Computer Aided Geometric Design 35–36 (2015) 215–224

Contents lists available at ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

High-quality topological structure extraction of volumetric data on C^2-continuous framework

Weisi Gu [a], Mei-e Fang [a,b,*], Lizhuang Ma [a]

[a] Department of Computer Science, Shanghai Jiaotong University, China
[b] Department of Computer Science, Hangzhou Dianzi University, China

ARTICLE INFO

Article history:
Available online 24 March 2015

Keywords:
Volumetric data
Box spline
Topological skeleton
Critical points

ABSTRACT

The existing approaches for topological structure analysis of volumetric data are mainly based on discrete methods, and the results usually need to be simplified and smoothened for further use. In this paper we propose a novel framework to extract the topology of volumetric data distinguished from the commonly-used piecewise linear framework. The data is reconstructed into a C^2-continuous quasi-interpolated space first by 7-directional box spline, where the value evaluation and differential calculations are both direct and accurate. Then Newton–Armijo method and homotopy continuation method are combined to solve critical points, and topological structures are extracted by connecting saddle-extremum arcs generated by a kind of numerical integral method. The parallel architecture of GPU is also applied to ensure the efficiency of our algorithms. A number of examples illustrate that our framework provides much smoother and clearer results compared with the piecewise linear framework.

1. Introduction

Volumetric data are the discrete representation of a 3D scalar field which are widely used in scientific researches, such as medical science, chemistry, materials science and other fields. The topology of volumetric data can be used to enhance visualization or help further scientific analysis. However, exploring the topology of discrete scalar fields is one of the most challenging problems in volumetric data studies.

Critical points are important topological features where the gradients vanished. For 3D data the critical points are classified into four types according to their indices, i.e. minima, 1-saddles, 2-saddles and maxima. Maxima and minima can be found easily in a discrete field, while the indices of saddles are not explicit. The discrete approaches depend on discrete Betti numbers to extract and classify critical points based on a piecewise-linear (PL) manifold. The isosurfaces of vertices are computed by marching cubes based methods, and the neighbour grids are partitioned into positive and negative connected components. A lookup table is used to determine whether the current vertex is a critical point and its index (Gerstner and Pajarola, 2000). When the marching cubes method is extended to trilinear interpolant (Nielson, 2003; Carr and Max, 2010), a similar lookup table is also used to get critical points in the trilinear field (Weber et al., 2002). All these methods are limited to isolated critical points, while many volumetric data models have constant areas, which

* Corresponding author.
E-mail addresses: srlzcc@sjtu.edu.cn (W. Gu), fme@hdu.edu.cn (M.-e Fang), ma-lz@cs.sjtu.edu.cn (L. Ma).

http://dx.doi.org/10.1016/j.cagd.2015.03.004
0167-8396/

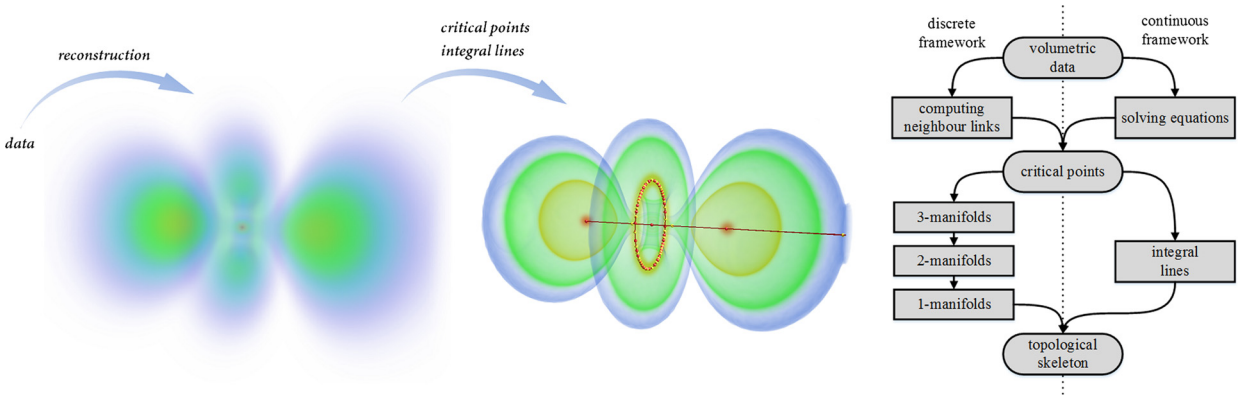


Fig. 1. Left: an example of extracting the topological skeleton by the method proposed in this paper. The first image shows the direct volume rendering result of Hydrogen data; the second image shows the skeleton formed by 2-saddle-maximum arcs (the red spheres are maxima and the yellow ones are 2-saddles), and the enhanced volume rendering result according with transfer function designed by the extracted critical values; Right: the comparison graph of different process flows between the existing discrete framework and the proposed continuous framework.

lead to critical regions. Weber et al. (2003) connect two graphs to represent the positive-connected and negative-connected neighbourhoods of a constant region, and classify regions according to the number of connected components in both graphs.

Many topology-related visualization and scientific analysis methods are based on the relationship between critical points. Contour tree abstracts the connectivity of level sets, which merge or split only at critical isovalues (Carr et al., 2003). Reeb graph extends contour tree that allows loops, and becomes another effective tool to analyze the topology of volumetric data (Pascucci et al., 2008). Some researchers devote their efforts to simplifying the topology (Carr et al., 2010) and generalizing the representation to multivariate fields (Carr and Duke, 2014).

Morse theory is a basic tool to define the relationship between critical points (Milnor, 1963). Morse–Smale (MS) complexes partition a data field into regions where the interior gradient flows are uniform (Edelsbrunnery et al., 2003). The discrete Morse theory combines CW complexes to form the quasi-Morse–Smale complex (Forman, 1998). Common approaches are based on PL framework. The main idea is to march from maxima and minima to form the ascending and descending manifolds, and then the MS complex is computed by intersecting these manifolds (Edelsbrunner et al., 2001, 2003). Due to the noise in the data and rounding off, there are many redundant critical points extracted by the former methods. Therefore the MS complex usually needs to be simplified to reflect the topology. Simplification of MS complex is done via removing insignificant critical points and merging the manifolds (Gyulassy et al., 2005). Later methods present new data structures and algorithms to speedup the simplification process and make the algorithms practical for real applications (Gyulassy et al., 2006, 2007, 2008). However the entire MS complex is still too complicated for visualization or analysis. Most of the time the purpose of computing the MS complex is to extract a summary skeleton of the volumetric data. The skeleton can be visualized directly to reflect the topology (Beketayev et al., 2011; Correa et al., 2011), or provides information to enhance the volume rendering result via improving the transfer function (Takahashi et al., 2004). The geometric features of the skeleton are also used for further analysis, for example, the structures of protein molecules can be learnt from the MS complex skeleton (Bajaj et al., 2012).

Although PL framework has been well studied and the performance is acceptable, there are some limitations of discrete approaches. The critical points extracted by discrete methods are always on the sample lattice, so the positions of critical points are not accurate for some low-resolution models. The algorithm of computing MS complex is difficult to implement, and the simplified result still hardly reflects the structure of scanned data with heavy noise. Besides, because the complex is defined on CW complexes, the borders of the manifolds are not smooth. Thus the gradient path from one critical point to another in discrete MS complex is along the sample vertices, which makes the visual result zigzag. Weinkauf et al. (2010) apply a bi-Laplacian filter to reconstruct a C^1-continuous scalar field to issue the problem. Norgard and Bremer (2013) compute the MS complex on a bilinear interpolated field so that the framework is continuous. However these methods are only designed for 2D scalar fields where the topology of saddles is simple.

To address these problems, we attempt to propose a unified framework to apply continuous methods on discrete volumetric data, though the topology-related computation on a continuous space is known as impractical because the critical points are difficult to solve (Forman, 1998). The framework is supposed to extract topological features with high accuracy in practical running time. Morse theory works for spaces with C^2-continuity, so the reconstructed data model should have the same or higher continuity. The basic tool to reconstruct the C^2-continuous scalar field is the 7-directional box spline proposed by Entezari and Möller (2006). This spline is often used for isosurface visualization due to the good spherical symmetry of its convolution kernel. We extend Kim and Peter's fast evaluation algorithm (Kim and Peters, 2008) to the topological computation and implement the critical points solver on GPU. The topological skeleton is connected by integral lines, which can be naturally computed on the C^2-continuous model. Fig. 1 illustrates the process flow of the framework.

Different from the discrete framework which uses combinatory structures of neighbour vertices to detect topological features, our framework solves critical points directly and skips the manifolds intersection to build the skeleton. The gradient

path is the real integral line in the space which is smooth and accurate. Moreover, the interpolation of the spline smoothen the data and the unsimplified result is fairly good to reflect the topology of the data.

The details of C^2-continuous reconstruction are discussed in Section 2. Some preprocessing operators are introduced to reduce computational cost and improve the quality of the result. In Section 3 we show how critical points are extracted by solving some polynomial systems within our framework. We also construct the directed gradient flow graph to represent the topological skeleton via connecting integral lines between critical points. These computations are explained in Section 4. At last we give the results and discuss the efficiency of our framework in Section 5.

2. Model reconstruction

Because solving critical points on a C^2-continuous field is the most time-consuming step, the interpolation method needs to be chosen carefully. The 7-directional box spline is the basic tool to reconstruct the C^2-continuous model in our framework. The polynomial form of the interpolation has lower degree than the tri-cubic B-spline so that it is easier to solve all critical points in later steps. The quasi-interpolation method based on box splines is mainly applied in visualizing isosurfaces of volumetric data (Entezari and Möller, 2006; Kim et al., 2008; Kim, 2013). We will demonstrate that this tool is useful for topological analysis as well. We first review the definition and properties of 7-directional box splines in Section 2.1, and then we introduce a pre-processing step that removes degeneracy for the later reconstruction in Section 2.2. The details of evaluation are explained in Section 2.3.

2.1. Box spline

A box spline M_{Ξ} is specified by a set of $n \geq s$ vectors $\{\boldsymbol{\xi_1}, \boldsymbol{\xi_2}, \ldots, \boldsymbol{\xi_n}\}$ in $\mathbb{R}^s$, which are the columns of matrix $\boldsymbol{\Xi}$. The spline is nonzero at point $\boldsymbol{x}$ only if $\exists \boldsymbol{t} \in [0,1)^n$ that $\boldsymbol{x} = \boldsymbol{\Xi t}$, which means the support of the box spline $\mathrm{supp} M_{\Xi}$ is the convex combination of these n vectors.

In the simplest case $n = s$, the box spline is defined as the characteristic function of its support:

$$M_{\Xi}(\boldsymbol{x}) = \begin{cases} \dfrac{1}{|\det \boldsymbol{\Xi}|} & \text{if } \boldsymbol{x} \in \mathrm{supp} M_{\Xi} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

When $n > s$, the spline is the convolution of additional $n - s$ vectors with the basic spline constructed by s vectors. We have the following recursive definition:

$$M_{[\Xi, \xi]}(\boldsymbol{x}) = \int_0^1 M_{\Xi}(\boldsymbol{x} - t\boldsymbol{\xi})\, \mathrm{d}t \tag{2}$$

The box spline is a piecewise polynomial with degree less than or equal to $n - s$ on $\mathrm{supp} M_{\Xi}$. If there are at most m vectors in columns of $\boldsymbol{\Xi}$ that do not span $\mathbb{R}^s$, the spline forms a C^{n-m-2}-continuous function. The proofs of the properties can be found in de Boor et al. (1993).

In this paper we only focus on 3D volumetric data, thus $s = 3$. We adopt the 7-directional box spline proposed in Entezari and Möller (2006). The spline matrix $\boldsymbol{\Xi}$ is given as the following:

$$\boldsymbol{\Xi} = \begin{bmatrix} 1 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 1 & 0 & -1 & 1 & -1 & 1 \\ 0 & 0 & 1 & -1 & -1 & 1 & 1 \end{bmatrix} \tag{3}$$

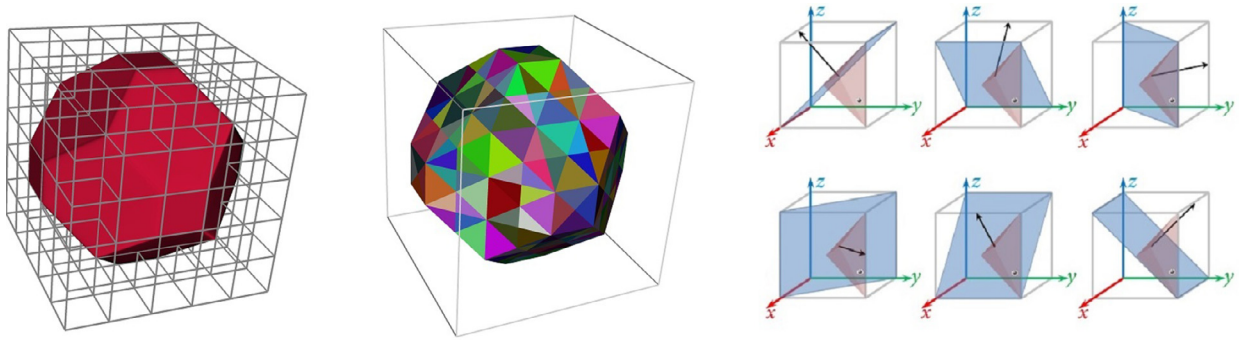
In this case, the support of the spline is a truncated rhombic dodecahedron embedded in $\left[-\frac{5}{2}, \frac{5}{2}\right]^3$ (see Fig. 2a). The entire support set is joined by 1272 tetrahedra (see Fig. 2b) and the spline function is a 4-degree polynomial in each tetrahedron. Due to that there are at most 3 vectors that do not span $\mathbb{R}^3$ (for example $(0,0,1)^T, (1,-1,-1)^T$ and $(-1,1,-1)^T$), the function is C^2-continuous.

Given a discrete 3D volumetric data set with a scalar function $f : \mathbb{Z}^3 \mapsto \mathbb{R}$ defined on it, we can reconstruct the continuous model via quasi-interpolation:

$$F(\boldsymbol{x}) = \sum_{\boldsymbol{j} \in \mathbb{Z}^3} f(\boldsymbol{j}) M_{\Xi}(\boldsymbol{x} - \boldsymbol{j}). \tag{4}$$

2.2. Pre-processing

The topology of the background is always not concerned in applications, so we label background regions first and do not search critical points in them to reduce the computational load. The background regions are commonly zero in simulated



(a) (b) (c)

Fig. 2. (a) The support set of the 7-directional box spline. (b) The tetrahedra that join as the support of the 7-directional box spline. (c) Six plains partition a unit cell into 24 tetrahedra.

data, but for some scanned data the noise makes the background nonzero. After setting a threshold to discard these nonzero background points, we set zero points on the border of the data set as the seeds of background regions. Then we traverse background points in breadth-first order and add the zero neighbour to the region.

The sample values of many volumetric data are nonnegative integers. Due to rounding off, the data sometimes have constant regions. From Eq. (4) we know that the interpolated values are constant if the sample values in the support of the box spline are constant. The index of a critical point is calculated by the eigenvalues of its Hessian matrix, so the constant regions should be removed to make the Hessian matrix non-degenerate. In PL framework, a symbolic perturbation is used to remove degeneracy (Gyulassy et al., 2007). Though we can do the same to the sample points in the supports of potential degenerate critical points, proving a general piecewise polynomial has no degenerate critical points is a difficult task. Our experiment shows that the degenerate critical points hardly appear in non-constant regions in the interpolated field for real volumetric data. Thus we just apply a filter operator μ to remove non-zero constant regions.

$$F(\boldsymbol{x}) = \sum_{\boldsymbol{j}\in\mathbb{Z}^3} \mu f(\boldsymbol{j}) M_{\Xi}(\boldsymbol{x}-\boldsymbol{j}) \tag{5}$$

This filter is supposed to smooth the discrete scalar function and make the center of a constant region be the critical point and other points regular. In our implementation μ consists of a major identity operator plus a minor Gaussian filter.

$$\mu f(\boldsymbol{j}) = \sum_{\boldsymbol{k}\in[-l,l]^3} g(\boldsymbol{k}) f(\boldsymbol{j}+\boldsymbol{k}) \approx f(\boldsymbol{j}) \tag{6}$$

where

$$g(\boldsymbol{k}) = \begin{cases} \varepsilon e^{-\frac{\|\boldsymbol{k}\|^2}{2}} & \text{if } \boldsymbol{k}\neq\boldsymbol{0} \\ 1-\sum_{\boldsymbol{k}\neq\boldsymbol{0}} g(\boldsymbol{k}) & \text{if } \boldsymbol{k}=\boldsymbol{0} \end{cases} \tag{7}$$

ε is a small constant term so that the filtered value approximates the original one. To ensure there are no constant regions left, the radius of the convolution kernel l should be greater than the radius of the largest inscribed sphere of constant regions in the interpolated field, for which 3 is enough in our implementation for most models. Fig. 3 gives a 2D example of how the filter works to remove degeneracy. The size of the support of 2D box spline is 3×3, if there is a non-zero constant region with larger size, the interpolated model would still include a smaller non-zero constant region in the center of the original constant region. The filter μ adjusts the values in constant regions and usually leaves a single extremum in the center.

2.3. Evaluation

Let $\bar{\boldsymbol{x}}$ be the nearest grid point of $\boldsymbol{x}$, and $\boldsymbol{\chi} = (\boldsymbol{x}-\bar{\boldsymbol{x}}) \in \left[-\frac{1}{2},\frac{1}{2}\right)^3$. Eq. (5) can be rewritten as the following:

$$F(\boldsymbol{x}) = \sum_{\boldsymbol{j}\in S} \mu f(\bar{\boldsymbol{x}}-\boldsymbol{j}) M_{\Xi}(\boldsymbol{\chi}+\boldsymbol{j}) \tag{8}$$

where $S = [-2,2]^3 \cap \mathbb{Z}^3$.

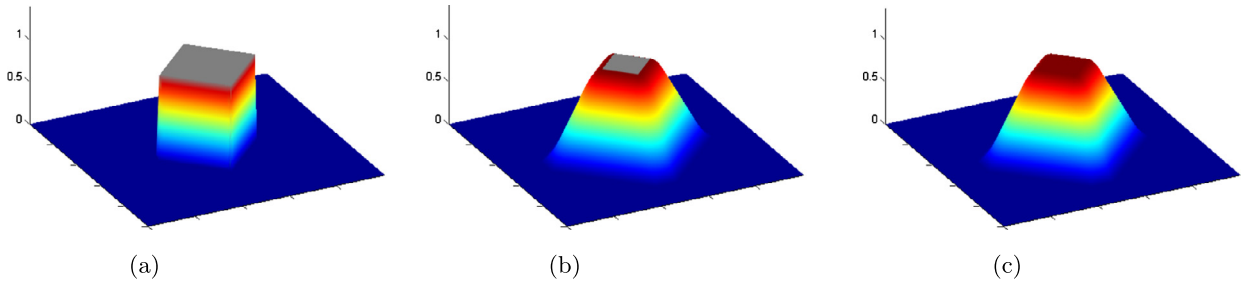


Fig. 3. The original function (a) has a constant region with scalar value equals to 1 colored in gray, so the interpolated function still may have constant regions as illustrated in (b). While after applying our filter, no constant region remains in (c), and the data is similar to that in (b). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The primitive method to compute M_{Ξ} is given by de Boor (1993). Kim and Peters (2008) prove that the polynomial pieces of a box spline can be represented in BB-form (Bernstein–Bézier-form) with coefficients in $\mathbb{Q}$.

$$M_{\Xi}(\boldsymbol{\chi}+\boldsymbol{j}) = \sum_{\boldsymbol{\alpha}} c_{\boldsymbol{j},\boldsymbol{\alpha}} b_{\boldsymbol{\alpha}}[\beta_{\sigma}(\boldsymbol{\chi}+\boldsymbol{j})] \tag{9}$$

β_{σ} maps $\boldsymbol{\chi}+\boldsymbol{j}$ in simplex σ onto its barycentric coordinates (x, y, z, w) that $x+y+z+w=1$. $\boldsymbol{\alpha}$ is a 4-dimensional vector $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ that $\alpha_i \in \{0, 1, 2, 3, 4\}$ and $\sum \alpha_i = 4$ for $1 \le i \le 4$. There are $\binom{7}{3} = 35$ different $\boldsymbol{\alpha}$ from the definition. $c_{\boldsymbol{j},\boldsymbol{\alpha}}$ are rational coefficients, and $b_{\boldsymbol{\alpha}}$ are Bernstein basis polynomials:

$$b_{\boldsymbol{\alpha}}(x, y, z, w) = \frac{24}{\alpha_1!\alpha_2!\alpha_3!\alpha_4!} x^{\alpha_1} y^{\alpha_2} z^{\alpha_3} w^{\alpha_4} \tag{10}$$

Fig. 2c shows the six plains partitioning a unit cube into 24 tetrahedra. If we shift $\boldsymbol{\chi}$ by $\boldsymbol{j} \in \mathbb{Z}^3$, the new position locates at the same tetrahedron of the unit cube which contains $\boldsymbol{j}$. Hence the barycentric coordinates of $\boldsymbol{\chi}+\boldsymbol{j}$ are the same as those of $\boldsymbol{\chi}$. We denote these equivalent coordinates by $\boldsymbol{\beta}_{\boldsymbol{\chi}} = (\beta_1, \beta_2, \beta_3, \beta_4)$.

Substituting the BB-form into Eq. (8), we have the final evaluation function:

$$\begin{aligned} F(\boldsymbol{x}) &= \sum_{\boldsymbol{j}\in S} \left[\mu f(\bar{\boldsymbol{x}} - \boldsymbol{j}) \sum_{\boldsymbol{\alpha}} c_{\boldsymbol{j},\boldsymbol{\alpha}} b_{\boldsymbol{\alpha}}(\boldsymbol{\beta}_{\boldsymbol{\chi}}) \right] \\ &= \sum_{\boldsymbol{\alpha}} \left[\sum_{\boldsymbol{j}\in S} \tilde{c}_{\boldsymbol{j},\boldsymbol{\alpha}} \mu f(\bar{\boldsymbol{x}} - \boldsymbol{j}) \right] \boldsymbol{\beta}_{\boldsymbol{\chi}}^{\boldsymbol{\alpha}} \end{aligned} \tag{11}$$

where $\tilde{c}_{\boldsymbol{j},\boldsymbol{\alpha}} = \frac{24}{\alpha_1!\alpha_2!\alpha_3!\alpha_4!} c_{\boldsymbol{j},\boldsymbol{\alpha}}$ are constant rational coefficients, and $\boldsymbol{\beta}_{\boldsymbol{\chi}}^{\boldsymbol{\alpha}} = \beta_1^{\alpha_1}\beta_2^{\alpha_2}\beta_3^{\alpha_3}\beta_4^{\alpha_4}$. The constant coefficients are pre-computed by solving a $35 \times |S| = 4375$ dimensional linear system for each tetrahedron. Since there are 24 tetrahedra, we get $4375 \times 24 = 105\,000$ coefficients. These coefficients are stored in a big table to be accessed.

For each evaluation, we need to do 4375 multiplications and sum them up to get all coefficients of $\boldsymbol{\beta}_{\boldsymbol{\chi}}^{\boldsymbol{\alpha}}$ in Eq. (11). To reduce the computation cost, we mark all zero-constant coefficients that multiplying data points outside the support of the spline. There are $1272/(24 \times 125) = 42.4\%$ nonzero constant coefficients left.

To compute the first and second order gradients of $F(\boldsymbol{x})$, we need to know the partial derivative function on each dimension. Take $g_x(\boldsymbol{x}) = \dfrac{\partial F(\boldsymbol{x})}{\partial x}$ as an example:

$$g_x(\boldsymbol{x}) = \sum_{\boldsymbol{\alpha}} \left[\sum_{\boldsymbol{j}\in S} \tilde{c}_{\boldsymbol{j},\boldsymbol{\alpha}} \mu f(\bar{\boldsymbol{x}} - \boldsymbol{j}) \right] \frac{\partial \boldsymbol{\beta}_{\boldsymbol{\chi}}^{\boldsymbol{\alpha}}}{\partial x} \tag{12}$$

Noting that the barycentric mapping is a linear transform, the right partial derivative part of Eq. (12) is a polynomial of $\boldsymbol{x}$ with degree less than or equal to 3. Rewriting g_x in BB-form, the number of Bernstein bases is $\binom{6}{3} = 20$. This means that we have 2500 constant coefficients to compute the partial derivative value for each tetrahedron on each dimension. For the second order gradients, the BB-form polynomial has $\binom{5}{3} = 10$ terms for each entry in the Hessian matrix. We pre-compute and store these constant coefficients together with those of $F(\boldsymbol{x})$.

3. Extracting critical points

A point in a scalar field is a critical point if its gradient is zero. A critical point is non-degenerate if its Hessian matrix is non-singular. The *index* of a critical point is the number of negative eigenvalues of its Hessian matrix. For 3D volumetric

data, the indices classify critical points into four types. Indices equal to 0, 1, 2, 3 for minima, 1-saddles, 2-saddles, and maxima respectively.

For a C^2-continuous model reconstructed by our method, a point $\boldsymbol{x}$ is a critical point means that $\boldsymbol{x}$ is the root of polynomial piece in the tetrahedron where $\boldsymbol{x}$ locates. Therefore we take two steps to extract all critical points. In the first step, we pick out the tetrahedra that may contain one or more critical points. In the second step, we solve the polynomial systems in these candidate tetrahedra to find out the accurate critical points.

3.1. Candidate tetrahedra

Supposing that the size of input volumetric data is $L \times W \times H$, the total number of tetrahedra is $24 \times L \times W \times H$. Because solving the polynomial systems is not fast enough for so many cases, we need to pick out candidate tetrahedra by discarding those impossibly containing critical points. The picking procedure needs to be fast and stable.

Let $\boldsymbol{\xi}$ be one column vector in $\boldsymbol{\Xi}$. The directional derivative of $M_{\boldsymbol{\Xi}}$ to $\boldsymbol{\xi}$ is the backward difference of $M_{[\boldsymbol{\Xi}\setminus\boldsymbol{\xi}]}$.

$$D_{\boldsymbol{\xi}} M_{\boldsymbol{\Xi}}(\boldsymbol{x}) = M_{\boldsymbol{\Xi}\setminus\boldsymbol{\xi}}(\boldsymbol{x}) - M_{\boldsymbol{\Xi}\setminus\boldsymbol{\xi}}(\boldsymbol{x}-\boldsymbol{\xi}), \tag{13}$$

$$\begin{aligned} D_{\boldsymbol{\xi}} F(\boldsymbol{x}) &= \sum_{\boldsymbol{j}\in S} \mu f(\bar{\boldsymbol{x}}-\boldsymbol{j}) D_{\boldsymbol{\xi}} M_{\boldsymbol{\Xi}}(\boldsymbol{\chi}+\boldsymbol{j}) \\ &= \sum_{\boldsymbol{j}\in S} \nabla_{\boldsymbol{\xi}} \mu f(\bar{\boldsymbol{x}}-\boldsymbol{j}) M_{\boldsymbol{\Xi}\setminus\boldsymbol{\xi}}(\boldsymbol{\chi}+\boldsymbol{j}) \end{aligned} \tag{14}$$

$\nabla_{\boldsymbol{\xi}}$ is a difference operator, such that $\nabla_{\boldsymbol{\xi}} \mu f(\boldsymbol{x}) = \mu f(\boldsymbol{x}) - \mu f(\boldsymbol{x}-\boldsymbol{\xi})$. $D_{\boldsymbol{\xi}} F(\boldsymbol{x})$ is the partial derivative function with respect to x, y, z when $\boldsymbol{\xi}$ is $\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z$ respectively.

Noting that the box spline is nonnegative, if the scalar values of the control points around $\boldsymbol{x}$ are monotonic along a direction, the backward difference will have the same sign. In this case, the gradient function is always positive or negative and has no zero point. Therefore a tetrahedron σ contains critical points only if the scalar values of its control points are not monotonic. We pick these tetrahedra as the candidates and solve the derivative polynomials to find critical points.

In Section 2.2, we have clarified that background points are usually insignificant for topological analysis. Thus all tetrahedra in background unit cubes are rejected.

3.2. Newton solver

In this section, we solve the derivative polynomials in tetrahedron σ. The three derivative polynomials are in BB-form as Eq. (12). The coefficients of the BB terms, i.e. the part in the square brackets of Eq. (12), are independent of the barycentric coordinates. Since the barycentric coordinates have three linear independent dimensions, we choose three of them $(\beta_1, \beta_2, \beta_3)$ and rewrite the four-variant BB-form polynomial into a ternary complete polynomial. We compute all the coefficients of the ternary complete polynomial to determine the exact representation of the derivative functions. Denoting the three gradient polynomials by g_x, g_y and g_z respectively, we have a ternary polynomial system. The solution of the polynomial system is a critical point only if it locates in the current tetrahedron, which means the solution should be nonnegative and the sum of three variants is less than or equal to 1. Thus finding critical points in a tetrahedron is equivalent to solving the following constrained polynomial system:

$$\begin{cases} g_x(\beta_1, \beta_2, \beta_3) = 0 \\ g_y(\beta_1, \beta_2, \beta_3) = 0 \\ g_z(\beta_1, \beta_2, \beta_3) = 0 \\ \text{s.t.} \quad \beta_1, \beta_2, \beta_3 \geq 0, \beta_1 + \beta_2 + \beta_3 \leq 1 \end{cases} \tag{15}$$

Our experiment shows that the polynomial systems in most candidate tetrahedra are inconsistent, and a few have more than one solution. To speedup the solving procedure, we first use a general numerical method to check whether the polynomial system has at least one solution. Newton–Armijo method is applied due to its fast convergence speed for smooth equations (Kelley, 2003). All candidate tetrahedra are streamed to a multithread Newton solver as the polynomial systems are independent of different tetrahedra. The initial guess for the solver of each system is set as the barycenter of the tetrahedron. Because the memory access is random and frequent when computing the coefficients of gradient polynomials, we implement this step on CPU. Once we find one available solution, the next step is to check whether the system has multiple solutions.

3.3. Homotopy continuation

There are two common approaches which can be used to solve all solutions of a polynomial system. One is the algebraic approach which computes the Gröner basis and performs eliminations iteratively to get all solutions, but the coefficients of

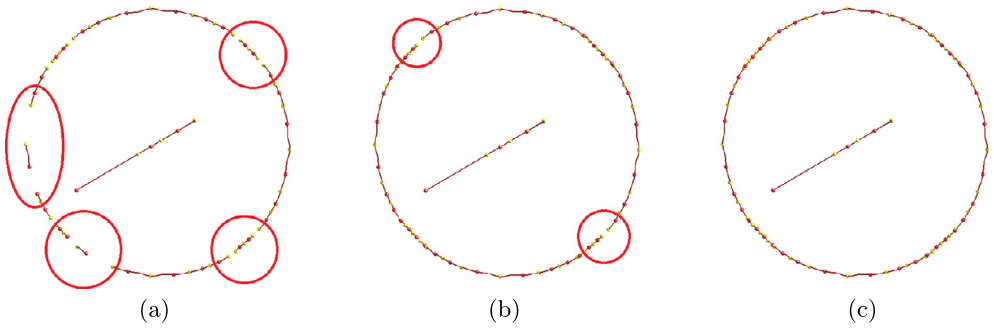


Fig. 4. Critical points extracted from Hydrogen model by (a) no homotopy continuation; (b) homotopy continuation with step length = 1/1000; (c) homotopy continuation with step length = 1/3000.

the polynomial must be rational. Our preprocess filter μ destroys the rationality of polynomial coefficients, so we adopt the other homotopy continuation method to solve the system.

Homotopy continuation is a general method to solve a polynomial system. Suppose that P is a polynomial system of n polynomial equations $P_1, P_2, \ldots, P_n$ in n unknowns, the number of its solutions is no more than $\prod d_i$, where d_i is the degree of P_i (Li, 1997). To solve the polynomial system P, a simple system with the same number of polynomials and variables is defined as Q. Q has exactly $\prod d_i$ isolated solutions and all of them can be solved easily. One typical suggestion of Q is:

$$\begin{cases} x_1^{d_1} = a_1 \\ \quad \vdots \\ x_n^{d_n} = a_n \end{cases} \tag{16}$$

$a_1, \ldots, a_n$ here are nonzero complex numbers that can be chosen randomly or manually. Because the solutions of the system $(1-t)Q + tP$ move smoothly as t varies from 0 to 1, we solve the problem by tracking the paths of the solutions from the initial system Q at $t = 0$ to the final system P at $t = 1$.

In our case, we have three ternary polynomials each with degree 3, which means there are at most 27 isolated roots in the entire complex space. If we use other interpolation whose polynomial form has degree more than 4, the number of possible isolated roots would be too large, which heavies the computational load. Assuming that the previously-used Newton iteration has solved one real solution (b_1, b_2, b_3) in the tetrahedron, we design Q as the following:

$$\begin{cases} \beta_1^3 = b_1^3 \\ \beta_2^3 = b_2^3 \\ \beta_3^3 = b_3^3 \end{cases} \tag{17}$$

Therefore (b_1, b_2, b_3) is always one of the solutions during the tracking process, and will not get lost if the homotopy continuation approaching is too fast. However, our experiment shows that many critical points are on the border of the tetrahedron. In this case, some components of the barycentric coordinates are zero, and $\beta_i^3 = 0$ is not an available initial polynomial because it has only one isolated solution. To address this problem, we adjust Q_i into $\beta_i^3 + \beta_i = 0$ so that Q has all 27 isolated solutions. However, the number of isolated solutions of P is usually less than Q. Some solutions merge to one during tracking, some diverge to infinite. Traditional homotopy continuation for a high dimensional polynomial system takes much effort to remove extraneous paths, but it's unnecessary for our low dimensional case. We track all 27 solutions parallelly and check the constraint condition at last.

We implement the homotopy continuation method on GPU. A CUDA warp runs 32 threads parallelly, so each polynomial system to be solved can be packaged into one block, and each thread tracking one solution. We store coefficients of temperate approaching polynomial system $(1-t)Q + tP$ in the shared memory together with P and Q. The tracking step length for t also influences the result. Shorter step length slows the approaching, but if the step length is too long, the current solution would be far away from the next solution on the continuation path, and some solutions may jump to another path. It results in missing some critical points. Fig. 4 shows the ring structures of Hydrogen model extracted by different homotopy continuation step lengths.

Our implementation set the step length to 1/1000, which is able to solve most multi-solution polynomial systems. Though there are still some critical points that may not be extracted for some special cases, the missing maxima can be found easily in later steps.

The index of a critical point is the number of negative eigenvalues of its Hessian matrix. Due to the C^2-continuity of our framework, the Hessian matrix of a critical point can be computed directly. All extracted critical points are classified into minima, 1-saddles, 2-saddles and maxima according to their indices.

Table 1
The running time of each step for different test data.

	Bucky $32 \times 32 \times 32$	Neghip $64 \times 64 \times 64$	Fuel $64 \times 64 \times 64$	Hydrogen $128 \times 128 \times 128$	Vertebra (*threshold* = 1000) $256 \times 256 \times 256$
preprocessing	0.02 s	0.17 s	0.14 s	0.17 s	1.59 s
Newton solver	1.13 s	2.19 s	0.65 s	1.07 s	2.82 s
homotopy continuation	6.49 s	1.57 s	3.53 s	1.6 s	16.76 s
connecting skeleton	0.68 s	0.39 s	0.19 s	0.16 s	0.62 s

4. Topological skeleton

In Morse theory, the *ascending manifold* of a minimum u is itself together with regular points whose integral lines originate at u. Symmetrically, the *descending manifold* of a maximum v is itself together with regular points whose integral lines end at v. The *Morse–Smale complex* partitions the space into regions in which the integral lines share common origins and destinations.

However, displaying the entire Morse–Smale complex is a difficult task, and usually we are only interested in the structures associated with extrema. Correa et al. (2011) use the *directed gradient flow graph* to approximate Morse–Smale complex. For 3D volumetric data, there are two *directed gradient flow graph*s. $[0, 1]$ graph is the 1-saddles with arcs connecting to minima, and $[2, 3]$ graph is the 2-saddles with arcs connecting to maxima. In discrete methods, the 3-manifolds are first computed, and then 2-manifolds that separate 3-manifolds are obtained (Gyulassy et al., 2007). 1-manifolds are found in a similar way to 2-manifolds.

Unlike the discrete method, 1-manifolds can be constructed directly within our framework. Because 1-saddles usually connect to the background region, $[2, 3]$ graph is better to represent the structure of the data. To construct a $[2, 3]$ graph, we traverse all 2-saddles and connect the arcs with maxima. For each 2-saddle, the Hessian matrix H is computed first. H has one positive eigenvalue, and that means the integral lines are outwards on the positive and negative directions of its corresponding eigenvector. We compute the integral lines along two directions of this eigenvector until they reach the maxima or go out of the space. These two integral lines are added to the graph as two arcs of the current 2-saddle.

Given a compact 3-manifold $\mathbb{M}$ and a smooth map $f : \mathbb{M} \mapsto \mathbb{R}$ defined on it, we call f a *Morse function* if all critical points are non-degenerate and have different values. An integral line is a maximal path in $\mathbb{M}$ whose tangent vectors agree with the gradients of f. Each integral line ends when it reaches a critical point where the gradient disappears. In our framework, the gradients of the reconstructed scalar function F are smooth. If we build a steady velocity field by the gradients of F, the integral line coincides with a streamline of the velocity field.

An integral line can be computed by numerical integral within our C^2-continuous framework. Shorter step length generates smoother line but heavies the computational load. To balance the computational cost and the line quality, the step length of numerical integral is adaptive. The next step length is initialized by the smaller one of a maximum length l_0 and k times of the currently applied step size. We test the next point and shorten the step by a ratio $q < 1$ iteratively if the next point breaks the monotonicity of the integral line or the orientation of the gradient change more than a threshold θ. Table 1 shows that extracting critical points is the most time-consuming stage in our experiment. Therefore we prefer a smaller step length in our implementation ($l_0 = 1/40$, $k = 2$, $q = 0.618$ and $\theta = 45°$). If the maximum step l_0 is small enough, the next point is likely to be in the same unit cell of the current point. In this case the control points are invariant, so we copy the values of control points to a local array to avoid the 3D index addressing and increase the cache hit ratio. If the l_0 is smaller, the next point may locate in the same tetrahedron as the current point. We save the coefficients of the BB-terms in a buffer as well to avoid re-computing the coefficients.

5. Results

We test our method on several well-known volumetric data models. For each data, we extract the critical points and draw the topological skeleton. Fig. 5 compares the results between our framework and PL framework. The integral lines are quite smooth under our framework, so the skeleton structures are very clear, while those obtained by the discrete method in Gyulassy et al. (2007) are zigzag.

The computational complexity of the discrete framework to build the skeleton is related to the number of sample vertices of volumetric data. While the performance of our approach mainly depends on the number of critical points in the reconstructed model because extracting critical points is the most time-consuming step. The box spline can be regarded as a filter of size $5 \times 5 \times 5$. Together with the preprocessing filter, the topology of volumetric data is simplified naturally after reconstruction, and the number of critical points is reduced.

The extraction of critical points is well parallelized in our framework. The Newton solver which screens polynomial systems to be further solved is streamed with 4 threads on Intel i5 CPU, while the homotopy continuation method is implemented on NVIDIA GT640. The running time for each step is listed in Table 1. The table shows that the performance is acceptable, and for simulated models which have little noise, our method is faster than the discrete method (the running time of computing discrete skeleton for Neghip and Hydrogen model is 7s and 27s respectively in Gyulassy et al. (2007)).

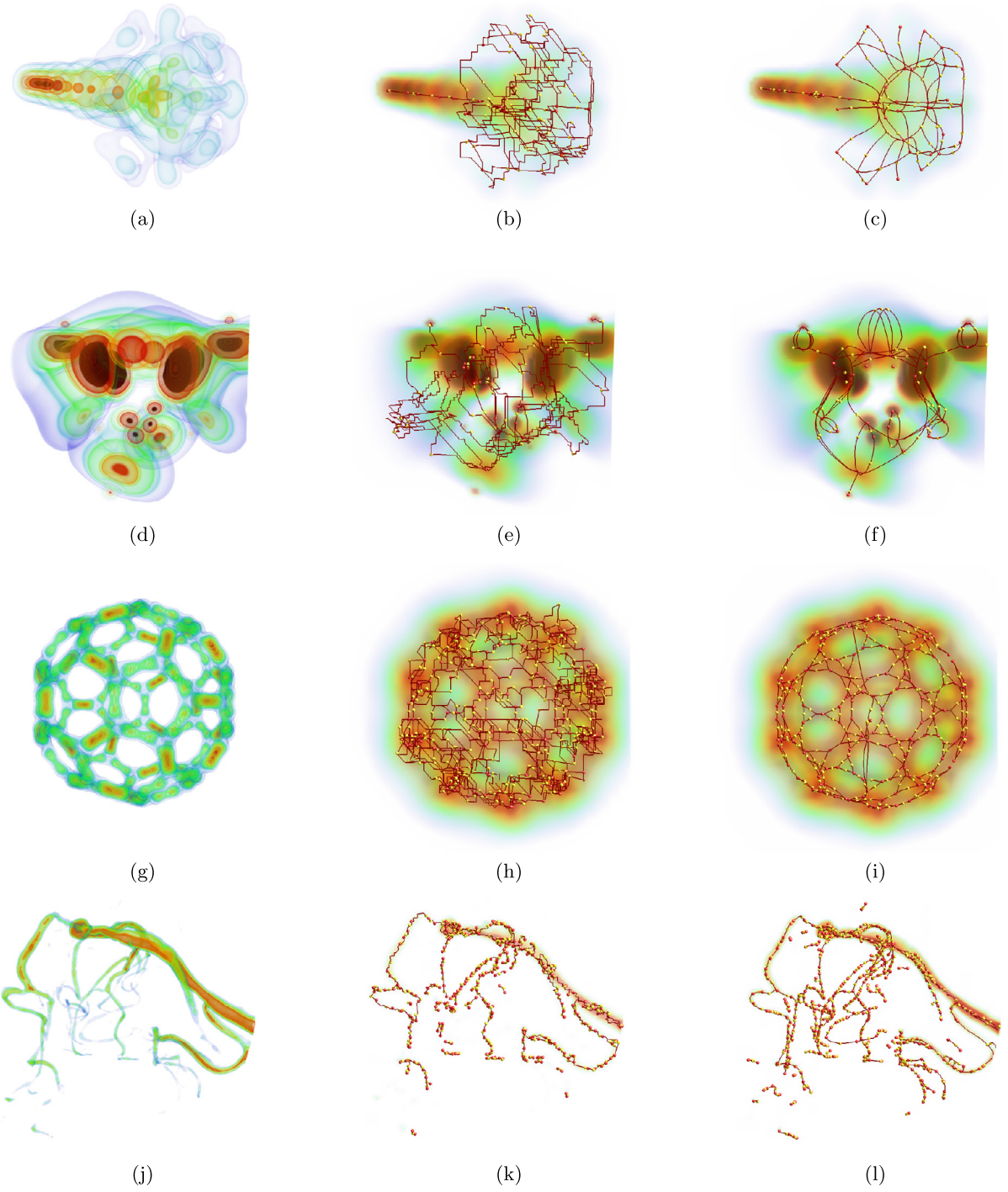


Fig. 5. The left column shows the volume rendering results of Fuel (a), Neghip (d), Bucky (g) and Vertebra (j) models with transfer functions customized by topological skeletons. The middle column shows the simplified 2-saddle-maximum arcs extracted by the discrete method proposed in Gyulassy et al. (2007). The right column shows the results extracted by our method.

6. Conclusion and future work

We presented a novel and practical framework that allows us to apply numerical methods for topological analysis on discrete volumetric data. Compared with the discrete methods, our approach can extract accurate critical points and generate smooth integral lines between critical points. The extracted skeleton is clear for visualization and can be simplified by traditional methods for further analysis. The framework is parallelism-friendly and has good performance.

However, for scanned data with large scale, high frequency noise makes the data bumpy in a local region. This phenomenon leads to the increasing number of critical points, which slows down the extraction and makes the skeleton complicated. Though we can apply simplification to the complex skeleton iteratively as the discrete approach does to remove the noise, solving critical points takes too much time. We are exploring new methods in the preprocessing step to remove high frequency noise, so that the number of candidate tetrahedra and the number of polynomial system to be solved are reduced to an acceptable level.

Acknowledgements

We are very grateful to the reviewers for their helpful suggestions and comments. The work described in this article is partially supported by National Science Foundation of China (61133009, 61272032), and Postdoctoral Science Foundation of China (2014M551413).

References

Bajaj, C., Goswami, S., Zhang, Q., 2012. Detection of secondary and supersecondary structures of proteins from cryo-electron microscopy. J. Struct. Biol. 177 (2), 367–381. http://dx.doi.org/10.1016/j.jsb.2011.11.032.

Beketayev, K., Weber, G.H., Haranczyk, M., Bremer, P.-T., Hlawitschka, M., Hamann, B., 2011. Topology-based visualization of transformation pathways in complex chemical systems. Comput. Graph. Forum 30 (3), 663–672. http://dx.doi.org/10.1111/j.1467-8659.2011.01915.x.

Carr, H., Duke, D., 2014. Joint contour nets. IEEE Trans. Vis. Comput. Graph. 20 (8), 1100–1113. http://dx.doi.org/10.1109/TVCG.2013.269.

Carr, H., Max, N., 2010. Subdivision analysis of the trilinear interpolant. IEEE Trans. Vis. Comput. Graph. 16 (4), 533–547. http://dx.doi.org/10.1109/TVCG.2009.10.

Carr, H., Snoeyink, J., Axen, U., 2003. Computing contour trees in all dimensions. Comput. Geom. 24 (2), 75–94. http://dx.doi.org/10.1016/S0925-7721(02)00093-7.

Carr, H., Snoeyink, J., van de Panne, M., 2010. Flexible isosurfaces: simplifying and displaying scalar topology using the contour tree. Comput. Geom. 43 (1), 42–58. http://dx.doi.org/10.1016/j.comgeo.2006.05.009.

Correa, C.D., Lindstrom, P., Bremer, P.-T., 2011. Topological spines: a structure-preserving visual representation of scalar fields. IEEE Trans. Vis. Comput. Graph. 17 (12), 1842–1851. http://dx.doi.org/10.1109/TVCG.2011.244.

de Boor, C., 1993. On the evaluation of box splines. Numer. Algorithms 5 (1), 5–23. http://dx.doi.org/10.1007/BF02109280.

de Boor, C., Höllig, K., Riemenschneider, S., 1993. Box Splines. Springer-Verlag.

Edelsbrunner, H., Harer, J., Zomorodian, A., 2001. Hierarchical Morse complexes for piecewise linear 2-manifolds. In: Proceedings of the Seventeenth Annual Symposium on Computational Geometry, pp. 70–79.

Edelsbrunner, H., Harer, J., Natarajan, V., Pascucci, V., 2003. Morse–Smale complexes for piecewise linear 3-manifolds. In: Proceedings of the Nineteenth Annual Symposium on Computational Geometry, pp. 361–370.

Entezari, A., Möller, T., 2006. Extensions of the Zwart–Powell box spline for volumetric data reconstruction on the Cartesian lattice. IEEE Trans. Vis. Comput. Graph. 12 (5), 1337–1344. http://dx.doi.org/10.1109/TVCG.2006.141.

Forman, R., 1998. Morse theory for cell complexes. Adv. Math. 134 (1), 90–145. http://dx.doi.org/10.1006/aima.1997.1650.

Gerstner, T., Pajarola, R., 2000. Topology preserving and controlled topology simplifying multiresolution isosurface extraction. In: Proceedings of the IEEE Visualization'00, pp. 259–266.

Gyulassy, A., Natarajan, V., Pascucci, V., Bremer, P.-T., Hamann, B., 2005. Topology-based simplification for feature extraction from 3D scalar fields. In: Proceedings of the IEEE Visualization'05, pp. 535–542.

Gyulassy, A., Natarajan, V., Pascucci, V., Bremer, P.-T., Hamann, B., 2006. A topological approach to simplification of three-dimensional scalar functions. IEEE Trans. Vis. Comput. Graph. 12 (4), 474–484. http://dx.doi.org/10.1109/TVCG.2006.57.

Gyulassy, A., Natarajan, V., Pascucci, V., Hamann, B., 2007. Efficient computation of Morse–Smale complexes for three-dimensional scalar functions. IEEE Trans. Vis. Comput. Graph. 13 (6), 1440–1447. http://dx.doi.org/10.1109/TVCG.2007.70552.

Gyulassy, A., Bremer, P.-T., Hamann, B., Pascucci, V., 2008. A practical approach to Morse–Smale complex computation: scalability and generality. IEEE Trans. Vis. Comput. Graph. 14 (6), 1619–1626. http://dx.doi.org/10.1109/TVCG.2008.110.

Kelley, C.T., 2003. Solving Nonlinear Equations with Newton's Method. SIAM.

Kim, M., 2013. Quartic box-spline reconstruction on the BCC lattice. IEEE Trans. Vis. Comput. Graph. 19 (2), 319–330. http://dx.doi.org/10.1109/TVCG.2012.130.

Kim, M., Peters, J., 2008. Fast and stable evaluation of box-splines via the BB-form. Numer. Algorithms 50 (4), 381–399. http://dx.doi.org/10.1007/s11075-008-9231-6.

Kim, M., Entezari, A., Peters, J., 2008. Box spline reconstruction on the face-centered cubic lattice. IEEE Trans. Vis. Comput. Graph. 14 (6), 1523–1530. http://dx.doi.org/10.1109/TVCG.2008.115.

Li, T.Y., 1997. Numerical solution of multivariate polynomial systems by homotopy continuation methods. Acta Numer. 6, 399–436. http://dx.doi.org/10.1017/S0962492900002749.

Milnor, J., 1963. Morse Theory. Princeton University Press.

Nielson, G., 2003. On marching cubes. IEEE Trans. Vis. Comput. Graph. 9 (3), 283–297. http://dx.doi.org/10.1109/TVCG.2003.1207437.

Norgard, G., Bremer, P.-T., 2013. Robust computation of Morse–Smale complexes of bilinear functions. Comput. Aided Geom. Des. 30 (6), 577–587. http://dx.doi.org/10.1016/j.cagd.2012.03.017.

Pascucci, V., Scorzelli, G., Bremer, P.-T., Mascarenhas, A., 2008. Robust on-line computation of Reeb graphs: simplicity and speed. ACM Trans. Graph. 26 (3), 58. http://dx.doi.org/10.1145/1276377.1276449.

Takahashi, S., Takeshima, Y., Fujishiro, I., 2004. Topological volume skeletonization and its application to transfer function design. Graph. Models 66 (1), 24–49. http://dx.doi.org/10.1016/j.gmod.2003.08.002.

Weber, G.H., Scheuermann, G., Hagen, H., Hamann, B., 2002. Exploring scalar fields using critical isovalues. In: Proceedings of the IEEE Visualization'02, pp. 171–178.

Weber, G.H., Scheuermann, G., Hamann, B., 2003. Detecting critical regions in scalar fields. In: Proceedings of the Symposium on Data Visualisation. VISSYM'03, pp. 85–94.

Weinkauf, T., Gingold, Y., Sorkine, O., 2010. Topology-based smoothing of 2D scalar fields with c1-continuity. In: Proceedings of the 12th Eurographics. IEEE–VGTC Conference on Visualization, EuroVis'10, pp. 1221–1230.