Available online at www.sciencedirect.com
ScienceDirect
Computer-Aided Design 39 (2007) 676–684
COMPUTER-AIDED DESIGN
www.elsevier.com/locate/cad

Controlled-topology filtering

Yotam I. Gingold*, Denis Zorin

Computer Science Department, New York University, 719 Broadway, 12th Floor, New York, NY 10003, United States

Received 22 March 2007; accepted 3 May 2007

Abstract

Many applications require the extraction of isolines and isosurfaces from scalar functions defined on regular grids. These scalar functions may have many different origins: from MRI and CT scan data to terrain data or results of a simulation. As a result of noise and other artifacts, curves and surfaces obtained by standard extraction algorithms often suffer from topological irregularities and geometric noise.

While it is possible to remove topological and geometric noise as a post-processing step, in the case when a large number of isolines are of interest there is a considerable advantage in filtering the scalar function directly. While most smoothing filters result in gradual simplification of the topological structure of contours, new topological features typically emerge and disappear during the smoothing process.

In this paper, we describe an algorithm for filtering functions defined on regular 2D grids with controlled topology changes, which ensures that the topological structure of the set of contour lines of the function is progressively simplified.

Keywords: Computational topology; Critical points; Filtering; Isosurfaces

1. Introduction

Many types of data are defined as scalar functions on unstructured or structured meshes. Such scalar fields are produced by MRI and CT scanners, scientific computing simulations, extracted from databases, or obtained by sampling distance functions to pointsets or surfaces. Quite often it is necessary to extract geometric information from such scalar fields, most commonly contour lines and isosurfaces, to which we will refer as *contours*. Contours often have to be extracted for multiple scalar function values, which motivates considering the topology of the complete set of contours, rather than that of an individual contour.

A variety of applications perform various types of processing either on the original scalar function data or on individual extracted isosurfaces. For example, the scalar field of an extracted contour can be smoothed to eliminate noise or to obtain a simplified representation of the object of interest, or enhanced to emphasize features of interest. The advantage of applying such processing operations to the scalar function, rather than to an extracted contour represented by a mesh, is that all contours are processed simultaneously and topology modification is possible. For example, spurious small-scale blobs due to noise in the scalar data can be eliminated.

While certain types of topology changes are desirable, other changes may have to be avoided. For example, if a blood vessel network is extracted from an image, breaking connected components of the isosurface is highly undesirable. Unfortunately, topological changes resulting from the application of a filter are difficult to control: even a simple Laplacian smoothing filter can result in undesirable disconnected components emerging (Fig. 8).

It is desirable to be able to control the topology changes occurring during the filtering process. In the extreme case, all changes can be disallowed, resulting in topology-preserving filtering; in other cases, certain types of topology changes are allowed while other changes are not. For example, if topological simplification is desired, merging components is acceptable while creating components is not.

One possible approach to the problem is to perform a complete topology analysis using contour trees or discrete Morse–Smale complexes, construct a topology hierarchy when topology simplification is desired, and design filters respecting the constraints (for example, maximal descent paths in the Morse–Smale complexes). The advantage of this approach is complete and entirely predictable control over topology. At

* Corresponding author. Tel.: +1 212 998 3473; fax: +1 212 995 4122.
E-mail addresses: gingold@mrl.nyu.edu (Y.I. Gingold), dzorin@mrl.nyu.edu (D. Zorin).

0010-4485/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cad.2007.05.010

the same time, the filter construction is far more complicated, as relatively complex constraints need to be imposed (cf. Bremer [1]). The other approach is to augment a filtering technique with topology control by detecting and preventing topological changes by local modifications to the filter. With the latter approach, one hopes that differences with the uncontrolled process can be minimized.

In this paper we describe an algorithm of the second type. Our algorithm adds topology control to flow-type filters which define a parametric family of results $p(t)$, $t = 0 \dots t_1$ where $p(0)$ is the vector of initial values of a scalar function defined on a two-dimensional regular grid. The idea of the algorithm is straightforward. The algorithm tracks critical points of the scalar function field to predict and determine the type of topology changes and locally adjust the rate of change of the scalar field to prevent disallowed changes. We consider three examples: Laplacian smoothing, sharpening, and anisotropic diffusion, and demonstrate that the algorithm makes it possible to control topology changes while retaining overall filter behavior and introducing relatively small errors.

Depending on problem semantics, the algorithm can either ensure complete topology preservation (e.g. for sharpening) or reduction in the number of topological features (for smoothing algorithms). The algorithm does not depend on dimension or structure of the grid in a fundamental way, and can be extended to three dimensions and unstructured grids.

2. Previous work

Analysis and simplification of the topology of vector fields and surfaces is a recurring topic in visualization, computational geometry and computer graphics. While many mesh simplification algorithms have allowed topology changes, in most cases these were unpredictable. One of the earliest examples of controlled topology simplification is He et al. [2], in which filtering on volume rasters is used to simplify objects. Alpha shapes were used in subsequent work [3]. These approaches use a reasonable algorithmic definition of topology simplification but do not track feature changes and focus on individual surfaces or solids enclosed by surfaces. Other work focusing on surfaces includes Guskov and Wood [4] and Wood et al. [5]. Our work is more similar to analysis and simplification of the structure of height fields and vector fields, for which complete collections of contours and stream lines are considered. The foundation of a significant fraction of recent work in this area is Morse theory (e.g. Milnor [6]), which relates the topology of smooth manifolds to critical points of functions defined on these manifolds. Helman and Hesselink [7] applied critical point analysis to flow visualization. de Leeuw and van Liere [8] was one of the first examples of topology simplification for vector fields. The simplification of de Leeuw and van Liere [8] is discrete rather than continuous: whole regions were removed from the field. An alternative approach was proposed in Tricoche et al. [9], which merges critical points into higher-order points. Tricoche [10] and Tricoche et al. [11] describe how topology can be continuously simplified by removing pairs of critical points.

Two important approaches to analyzing topological structure of a scalar field are contour trees [12–15] and structures based on Morse–Smale complexes [1,16–18].

The contour tree is a data structure that fully describes the topology of a scalar field, with contours passing through critical points as nodes. Contour trees have been extensively used for fast isosurface extraction and define a natural topological hierarchy. The 2D Morse–Smale complex has vertices at critical points which are connected by maximal descent paths: similar structures are also defined for an arbitrary number of dimensions. A topological hierarchy can also be defined using Morse–Smale complexes and feature *persistence*.

Both types of structures were used for topological simplification of scalar fields using associated hierarchies and different types of persistence functions, e.g. in recent papers [1, 19]. In both cases, the scalar field function values are updated to eliminate features locally. In Bremer et al. [1], smoothing is performed with constrained Morse–Smale complex boundaries, and the boundaries themselves are adjusted using smoothing.

While our algorithm can be used to construct topological hierarchies, this is not our primary goal. We aim to provide a tool that adds topology control to a variety of processing techniques for scalar data. Our primary concern is not visualization and analysis of a scalar field topology; rather, we aim to augment existing image processing tools with topological guarantees, while preserving the basic behavior of the tool.

The recent work by Sohn and Bajaj [20] on time-varying contour topology deals with similar types of evolving scalar data. The goal of this work is accurate feature tracking in a given dataset, while our goal is to alter a time-dependent dataset to eliminate certain types of topological events.

The topological evolution resulting from filtering, smoothing in particular, is often considered in vision and medical imaging literature. The concept of *scale space* based on Laplacian smoothing (heat flow), proposed in Witkin [21], is used in a variety of applications, and one can consider similar types of constructions based on different flows, such as anisotropic diffusion [22] or curvature flow. The topology of scale spaces was studied in Damon [23], from a mathematical point of view, and more recently in Florack and Kuijper [24].

3. Topological preliminaries

Our algorithm is based on the correspondence between topological features and pairs of critical points. *Topological events*, i.e. the changes in the topology of the sets of contours, are associated with changes in these pairs. For example, a new connected component appears if a pair of critical points appear, and an existing component vanishes when two critical points merge.

In this section, we briefly review relevant definitions and facts from differentiable and discrete topology, which are used by our algorithm.

Smooth Morse theory. We restrict our attention to the case of functions defined on the plane. A function f is called a Morse function if it is at least twice differentiable, its values at critical points defined by $\nabla f = 0$ are distinct and its *Hessian*, i.e. the matrix of second derivatives, has nonzero determinant at critical points. Critical points with nondegenerate Hessian are called *simple*. If the Hessian vanishes at a critical point, it is called *complex*.

The Morse lemma states that for a suitable choice of coordinates the function has the form $\pm x^2 \pm y^2$ in a neighborhood of any critical point. The number of minuses is the *index of the critical point*. Saddles have index 1, maxima have index 2, and minima have index 0. For functions defined on planar domains, it is convenient to add a point at infinity to the plane and assign a minimum to it with infinite negative value.

The indices of critical points x of functions defined on a sphere are known to satisfy

$$\sum_x (-1)^{\mathrm{index}(x)} = 2. \tag{1}$$

The quantity $(-1)^{\mathrm{index}(x)}$ is called the *topological charge* of a critical point.

Critical points can be used to describe the topological structure of the contour lines of the function f. The level set $f^{-1}(c)$ is a smooth curve unless c is a value at a critical point. Furthermore, if we consider an interval of values $[c_1, c_2]$ not containing any critical point's value, the level sets for all $c \in [c_1, c_2]$ have the same topology. Thus, the critical points define all topological changes of level set curves.

The singular level sets corresponding to maxima and minima consist of isolated points and correspond to vanishing/appearing features if we regard the traversal of increasing values of $f(x)$ as advancing in time. Saddles correspond to the merging/splitting of features.

Multiple simple, closed contours meet at a saddle. (Exactly two if the saddle is simple.) We define a feature to be the interior of one of these contours pairing an unpaired extremum in the interior with the saddle. The nesting structure of the feature contours implies a feature hierarchy. (There are many possible pairings and hence many possible feature hierarchies for a given topology.)

Discrete Morse theory. While one can define C^2 interpolants for functions especially on regular grids, studying critical points of such functions is difficult. It is preferable to generalize the notions of smooth Morse theory to piecewise-linear functions. We mostly follow Edelsbrunner et al. [17] in our definitions. A critical point of a piecewise linear function is always at a vertex. Its type can be inferred from comparing the values of the critical point with adjacent values. We consider *lower* and *upper* stars of a vertex. The lower star consists of all simplices incident to v whose vertex faces have function value $f(w) \le f(v)$, and the upper star consists of all simplices whose vertex faces have function value $f(w) \ge f(v)$. (Fig. 1, based on Edelsbrunner et al. [17]). Each star can be decomposed into continuous wedges. If one of the stars coincides with the entire neighborhood the point is a local minimum (upper star) or maximum (lower star). If each star has exactly one wedge, the point is considered regular. If the number of wedges in each

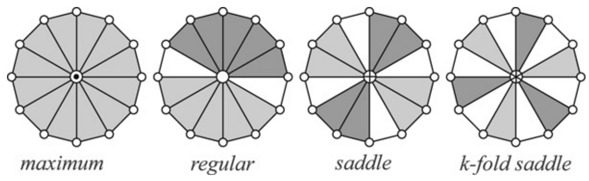

Fig. 1. Different vertex types [17].

star is $k + 1$ for $k \ge 1$, then the star is a k-fold saddle (simple saddle for $k = 1$). Unlike the smooth case, complex saddles are stable. The type of a vertex can be determined by the vertex *signature* i.e. a sequence of ones and zeros corresponding to the adjacent points, ordered counterclockwise, with one indicating that the value at the adjacent vertex is higher than the value at the center.

A discrete Morse function is any piecewise linear function for which the values at critical points are distinct. To operate on arbitrary piecewise linear functions, we require a tie-breaking scheme such as Simulation of Simplicity [25]. As we only use value comparisons in our algorithm, it is sufficient to simply assign an ordering based on indices of vertices to break ties. However, it is necessary to use first-order perturbations to resolve ties for calculated event times (Section 4).

Singularities of parametric families of functions. Smoothing a function using a flow equation, e.g. $\partial f / \partial t = \Delta f$, leads to a solution $f(x, t)$, which can be regarded as a one-parameter family of functions. While complex critical points of functions can be eliminated by small perturbations, this is no longer true for parametric families. To clarify this, consider any family of functions of one argument $f(x, t)$, such that $f(x, -1)$ has a maximum and a minimum and $f(x, 1)$ has no extrema. In the beginning, the derivative of the function has two roots, and at $t = 1$ it has no roots. Therefore, no matter what perturbation we use, there is a parameter value t_1 such that the derivative has exactly one root. One can easily see that this extremum cannot be generic: as the extrema merge, the Hessian is always positive at one and negative at the other. This implies that it is zero at the moment they merge.

By choosing a suitable coordinate system, a generic singularity with one parameter in one dimension can be reduced to the form

$$f(x, t) = x^3 + xt.$$

Stable complex singularities arising in parametric families of functions are studied in singularity theory. It turns out that in two dimensions a single parameter singularity has a similar form

$$x^3 + xt + ay^2. \tag{2}$$

One can see (Fig. 2) that it corresponds to two critical points (a saddle and a minimum) merging at a point (annihilation) for t increasing and a saddle–minimum pair appearing at a point (creation) for t decreasing. For $t = 0$ the critical point is always complex.

Discrete case. Next we consider the discrete analogs of creation and annihilation events. Because of the presence of complex

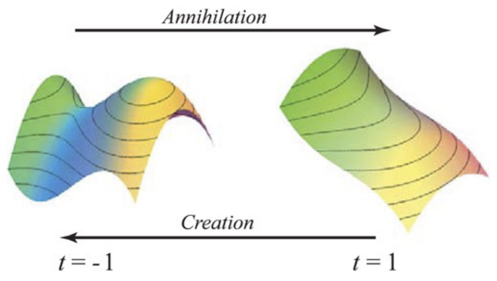

Fig. 2. Annihilation and creation events.

saddles, more event types are possible (Fig. 3). In addition, because of the piecewise linear nature of the functions, the singularities do not move continuously but in discrete jumps from one vertex to another. So we introduce one more event type corresponding to singularities changing locations.

We consider piecewise linear functions evolving piecewise linearly in time. The topological picture for such function changes discretely. Each change corresponds to a *value flip* at an edge (v, w), *i.e.* transition from configuration $f(v) > f(w)$ to the configuration $f(v) < f(w)$, with the relative order of all other adjacent functions remaining unchanged. Again, one can assume that two elementary value flip events never coincide in time by simple tie-breaking.

Merge/Annihilation. We use the term merge events to denote any events which result in a critical point disappearing. Due to the presence of k-fold saddles, many variations of events are possible, with only the one involving a simple saddle and a maximum or minimum resulting in annihilation. Other types of events include saddle merges and a maximum or minimum absorbed by a saddle with a change in the number of folds. We consider all these events admissible.

Creation. Similarly, creation events are the events resulting in creation of a critical point, and can be of many types. Only one type (emergence of a saddle–minimum/maximum pair) results in critical points created from a regular point. We forbid all events that involve the creation of a maximum or a minimum. We allow saddle separation, as we view k-fold saddles as k-simple saddles merged together.

Exchange. Adjacent k-fold saddles may exchange folds; no critical points move in this case.

Move. This type of event corresponds to the situation when a critical point vanishes at one end of an edge with a point of the same type appearing at the other.

Non-event. Some value flip events may result in no changes in the type of endpoints of the edge.

For each of our filter examples all event types are present for sufficiently complex images, unless topology control is applied.

Creation events in smoothing. Intuitively, one would think that at least for Laplacian smoothing no creation events can occur. However, it is known not to be the case, and indeed creation events can be observed if Laplacian smoothing is applied to real datasets (Fig. 4). A typical local configuration resulting in such events is shown in Fig. 5.

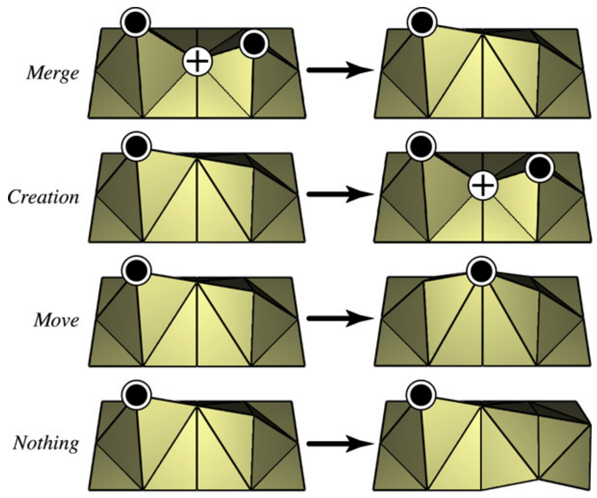

Fig. 3. Discrete topological event types resulting in changes in critical point locations.

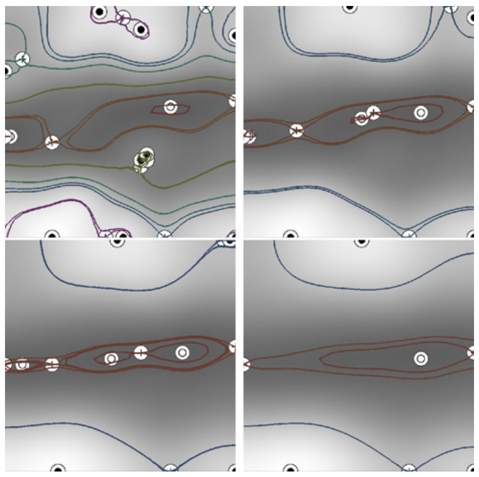

Fig. 4. A magnified fragment of the Puget Sound dataset, for several steps of Laplacian smoothing. Note a pair of critical points appearing, moving apart, and then merging.

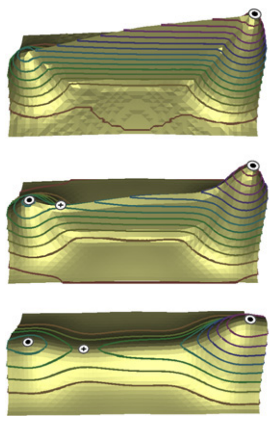

Fig. 5. A typical creation event for Laplacian smoothing: a thin ridge connecting two bumps. The new maximum-saddle point is quite stable and disappears only after the shorter bump is almost entirely smoothed out.

4. Algorithm

The algorithm operates on *scalar values* $p(v)$ defined at vertices $v \in V$ of a mesh which evolve over time. We assume that there is a monotonic ordering of values at any fixed time, breaking ties in a uniform way using vertex indices. We use data defined on regular grids in our examples, but the algorithm can be used for arbitrary meshes.

In addition to the input data, the user defines the set of *disallowed* topological events.

The filter, which is separate from the topology control algorithm, for a given time step Δt and data $p^l(v)$ corresponding to time t_l produces *proposed values* $\bar{p}^{l+1}(v) = F(p^l(v), \Delta t)$ for the moment $t^{l+1} = t^l + \Delta t$. The algorithm can be applied to any evolving scalar field. We have considered the following filter examples: linear diffusion (Laplace filtering), sharpening, and anisotropic edge-enhancing diffusion [22].

The choice of disallowed topological events depends on the semantics of the filter. When sharpening one wishes to exaggerate existing features, so all topological events are prevented. For the other two filters, we wish to reduce topological complexity by smoothing, so creation events are prevented. For diffusion, the updates are computed using either explicit or implicit time stepping, the latter allowing large time steps Δt.

A single step of the *outer loop* of the algorithm requests the proposed values $\bar{p}^{l+1}(v)$ from the filter and assumes linear evolution between $p^l(v)$ and $\bar{p}^{l+1}(v)$ for each value. The time of all possible topology events is computed, and the proposed values are adjusted to ensure that disallowed events do not occur. After adjustment, events need to be recalculated, and further adjustments may be necessary. The process is iterated until there are no disallowed events.

The algorithm may fail to produce progress, if for all points v computed update is below a threshold ε, which for any point v may be due to two reasons: either the local time step or $\bar{p}^{l+1}(v) - p^l(v)$ is too small, which means that the filter has converged to a limit value. The algorithm terminates if it either reached the target time, or failed to produce progress.

One iteration of the algorithm in detail. The algorithm uses two maps:

– *Critical point map* `status`(v) indicating if a vertex is a (discrete) maximum, minimum, saddle or regular point, and storing a critical point's ID. This map is initialized using the original data and incrementally updated at every step.
– *Value flip map* `flip`(v, w), where (v, w) is an edge, recording value flips observed at the current step, and the time for each flip (the timestamp).

Step 1: *Obtain proposed values.* Compute trial point positions $\bar{p}^{l+1}(v)$ using the uncontrolled filter:

$$\bar{p}^{l+1} = F(\Delta t, p^l(v)).$$

If no progress is made, i.e. the difference $|\bar{p}^{l+1} - p^l|$ does not exceed a user-defined threshold, the algorithm terminates.

Step 2: *Identify and sort the value flip events.* We regard the evolution of the mesh as linear between p^l and $\bar{p}^{l+1}$. This guarantees that each edge may flip value no more than once during a time step. For each edge (v, w) we determine whether or not it flips value during the trial step and store the result in `flip`(v, w).

Then for all value flip events we compute the exact time of the value flip `flip.time`(v, w), using linearity and our tie-breaking scheme, and sort the value flipping edges according to this time.

Let a, b, c, d be four points, such that the pairs $(p(a), p(b))$ and $(p(c), p(d))$ both change order between p^l and $\bar{p}^{l+1}$. We assign unique infinitesimal perturbation scales $e(a), e(b), e(c), e(d)$ to each vertex, i.e. we regard the value at a vertex $v \in \{a, b, c, d\}$ as a polynomial $p(v) + \varepsilon e(v)$ in ε.

Let t be the solution of $p^l(a)(1-t) + t\bar{p}^{l+1}(a) = p^l(b)(1-t) + t\bar{p}^{l+1}(b)$, and let t' be the solution of $p^l(c)(1-t) - t\bar{p}^{l+1}(c) = p^l(d)t - (1-t)\bar{p}^{l+1}(d)$. We assume that both solutions exist (otherwise there are no events that need to be ordered). Then

$$t = \frac{p^l(b) - p^l(a) + \varepsilon e(b) - \varepsilon e(a)}{(\bar{p}^{l+1}(a) - p^l(a)) - (\bar{p}^{l+1}(b) - p^l(b))}$$

and

$$t' = \frac{p^l(d) - p^l(c) + \varepsilon e(d) - \varepsilon e(c)}{(\bar{p}^{l+1}(c) - p^l(c)) - (\bar{p}^{l+1}(d) - p^l(d))}.$$

To compare t and t', we consider $f(\varepsilon) = t - t'$. If $f(0) > 0$ then no tie-breaking is necessary. However, it is possible that $f(0) = 0$. This is the case where perturbation is necessary: we break the tie by computing the first-order term of $f(\varepsilon)$, which determines the sign of $f(\varepsilon)$ for all sufficiently small $\varepsilon > 0$.

If $f(0) = 0$, then $f(\varepsilon)$ is given by

$$f(\varepsilon) = \varepsilon \left(\frac{e(b) - e(a)}{(\bar{p}^{l+1}(a) - p^l(a)) - (\bar{p}^{l+1}(b) - p^l(b))} \right. \\ \left. - \frac{e(d) - e(c)}{(\bar{p}^{l+1}(c) - p^l(c)) - (\bar{p}^{l+1}(d) - p^l(d))} \right).$$

Step 3: *Detect disallowed events.* Next, we traverse the event list ordered by time, detecting creation events. For each value flip event (v, w), we determine its type based on the changes of signatures of endpoints v and w in the critical point map for p^l and $\bar{p}^{l+1}$, as explained in Section 3.

If we find a disallowed event (v, w) at a time $t =$ `flip.time`$(v, w) < t_{l+1}$, we set

$$\bar{p}^{l+1}(v) \leftarrow (t - \delta)(\bar{p}^{l+1}(v) - p^l(v)) + p^l(v)$$

and

$$\bar{p}^{l+1}(w) \leftarrow (t - \delta)(\bar{p}^{l+1}(w) - p^l(w)) + p^l(w)$$

and return to Step 2. In so modifying $\bar{p}^{l+1}$, we set the proposed values for v and w to δ before the value flip along the line from p^l to $\bar{p}^{l+1}$.

Thus the value flip no longer occurs as the values of v and w evolve from p^l to $\bar{p}^{l+1}$. In other words,

$$\operatorname{sign}(p^l(v) - p^l(w)) = \operatorname{sign}(\bar{p}^{l+1}(v) - \bar{p}^{l+1}(w)).$$

This ensures that undesired topological events never occur. An example is shown in Fig. 6.

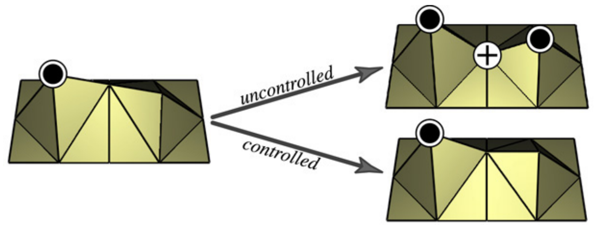

Fig. 6. Without topology control, a maximum–saddle pair is created. With topology control, the creation event is suppressed.

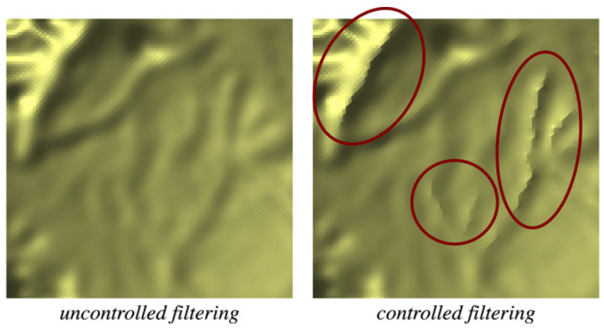

Fig. 7. Temporary local artifacts in the smoothing process due to topology control, magnified image.

If no creation events are found at the end of the event list traversal, then the inner loop is terminated, $p^{l+1}(v)$ is set to $\bar{p}^{l+1}(v)$, and the next increment is obtained from the filter.

We are guaranteed to exit the inner loop. Every time we disallow an event, the proposed values $\bar{p}^{l+1}$ approach the current values p^l. Because this process is monotonic, eventually $\bar{p}^{l+1}$ will not contain any disallowed events. This may occur because $\bar{p}^{l+1}$ no longer differs from p^l; a small difference $|\bar{p}^{l+1} - p^l|$ is a termination condition for the algorithm.

The pseudocode for the algorithm is as follows.

```
repeat
    for all $v \in V$ do
        $\bar{p}^{l+1}(v) \leftarrow F(\Delta t, p^l(v))$
    end for
    repeat
        $\overline{\mathtt{status}}^{l+1} \leftarrow \mathtt{status}^l$
        flip ← all value flipping edges $(v, w)$, sorted
            by time
        for all $f(v, w) \in$ flip do
            update $\overline{\mathtt{status}}^{l+1}(v)$ and $\overline{\mathtt{status}}^{l+1}(w)$
            if $f$ is a disallowed event then
                modify $\bar{p}^{l+1}(v)$, $\bar{p}^{l+1}(w)$
                break
            end if
        end for
    until no undesired event is found
    $p^{l+1} \leftarrow \bar{p}^{l+1}$
    $\mathtt{status}^{l+1} \leftarrow \overline{\mathtt{status}}^{l+1}$
    $l \leftarrow l + 1$
until no progress possible or target time reached
```

Depending on the type of the filter, the algorithm behaves in different ways: if locally the filter smooths the values, it still may create features. However, these features are short-lived, so the values frozen by the algorithm are likely to be released quickly (see the discussion of the Laplacian smoothing example in Section 5). On the other hand, if the filter is locally enhancing and tends to create new features, the algorithm will prevent it from altering the image, which we consider the desirable behavior in such cases.

The return to Step 2 after modification of the proposed values is essential, as the value modification results in rearrangement of critical points. This results in a considerable increase in time vs. simple application of a filter for initial steps. However, following a sufficiently large number of smoothing steps, few critical points interact with each other in a given step, so fewer inner cycles are necessary for each time advance.

5. Results

We show the results of our algorithm for three different filters:

– Discrete Laplacian smoothing (diffusion):
$$p_L^{l+1}(v) = p^l(v) + \Delta t \sum_{\text{edges}(v,w)} (p^l(w) - p^l(v)).$$
– Sharpening, for a fixed n,
$$p_S^{l+1}(v) = p^l(v) + \Delta t (p_L^n(v) - p^0(v)).$$
– Discrete anisotropic diffusion [22]:
$$p_{\text{AD}}^{l+1}(v) = p^l(v) + \Delta t \sum_{\text{edges}(v,w)} \frac{p^l(w) - p^l(v)}{1 + \|p^l(w) - p^l(v)\|^2/k^2}.$$

For diffusion, we have implemented both explicit and implicit time stepping, the linear systems solved using SuperLU [26] in the latter case.

Note that for the sharpening filter for any time t is a linear interpolation between $p^0(v)$ and $p_L^n(v)$.

We use two artificial datasets for comparison: a ridge, showing the emergence of a high-persistence saddle for Laplacian smoothing without topology control, and a pure noise image, which almost entirely consists of critical points in the beginning.

We use several real datasets: the Puget Sound terrain map, a CT scan slice of a cow brain, a retina image, and a CT scan slice of a human torso. In the images showing critical points, crosses denote saddles, empty circles denote minima, and circles with dots denote maxima.

Fig. 8 (right) compares the behavior of the Laplacian smoothing filter for artificial data with and without topology control. Note that while the topology change is prevented, there is little impact on the overall surface smoothness. Fig. 4 shows a similar event in the Puget Sound dataset.

Fig. 9 shows that even for very complex topologies (in the initial image almost every point is critical) the algorithm does not get stuck because of excessive numbers of frozen values.

The analysis of the topology of the scale space provides intuition into why this is the case: in Damon [23] it is shown that, unlike the general case, there is an asymmetry between creation and annihilation events in the evolving data resulting from Laplace smoothing. In this case, the two types of events can be described by different normal forms and further analysis

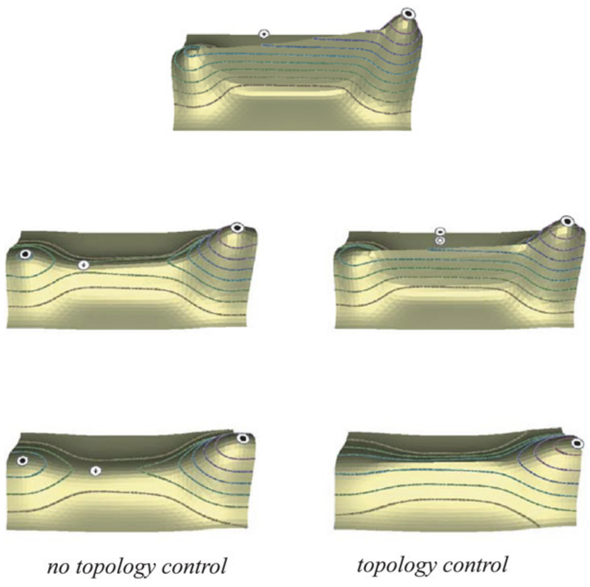

Fig. 8. Comparison of our topology controlled smoothing algorithm with uncontrolled Laplacian smoothing for a simple artificial dataset.

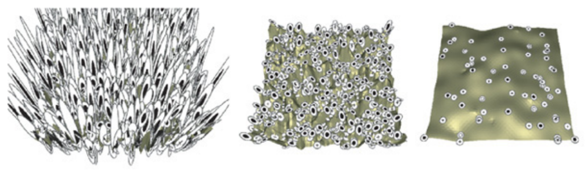

Fig. 9. Topology controlled Laplacian smoothing for random initial data.

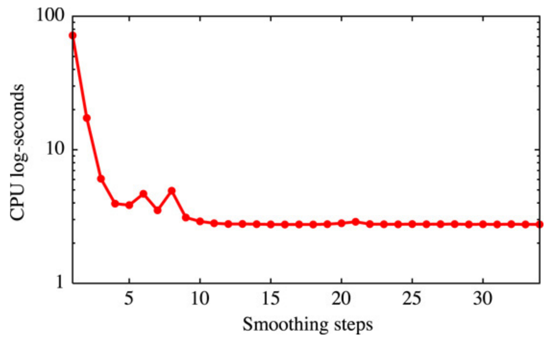

Fig. 10. Algorithm performance: CPU time per smoothing step, for implicit Laplacian smoothing, running on a Xeon 2.4 GHz.

shows that newly created critical points have long expected lifetime only on ridges, i.e. on a small subset of the image. Thus, on average, the lifetime of created topological features is short, and values need to move only slightly slower to avoid feature creation entirely.

Next, we compare the results for three filters (Fig. 13). For each filter we show five images: the original image, the filtered image without topology control, the filtered image with topology control, the relative magnitude of difference in each case, and the map of all created features without topology control. Note that most if not all created features are small, and hard to see in the image, and the filtering results in both cases are visually similar. We also observe that errors are small and localized for two filters (the maximal error is approx. 2% for Laplacian smoothing and 6% for anisotropic diffusion). While overall the changes introduced by topology control are small, for smoothing filters one can sometimes observe artifacts near values which are prevented from changing. In these areas the surface appears less smooth than in adjacent areas—for example, in the sharp ravines visible in the terrain in Fig. 13 (left column) shown magnified in Fig. 7. These artifacts exist for a relatively short time and are eliminated in subsequent filtering steps.

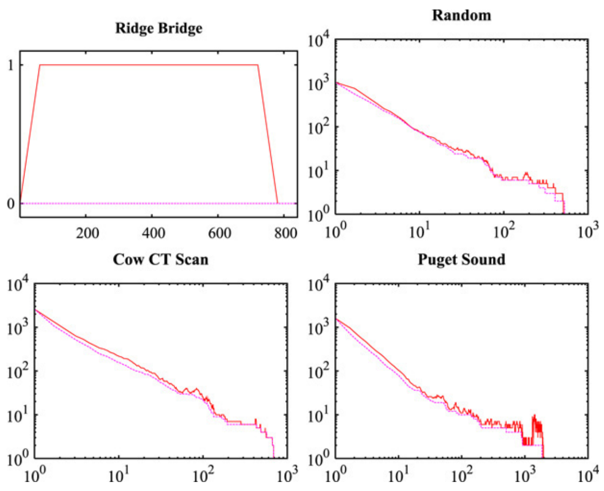

Fig. 11. The number of critical points (y-axis) as functions of the iteration number (x-axis) for four datasets undergoing Laplacian smoothing (red line) and topology controlled Laplacian smoothing (purple dotted line). Upper row: artificial datasets. Lower row: the cow brain CT scan and Puget Sound dataset. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

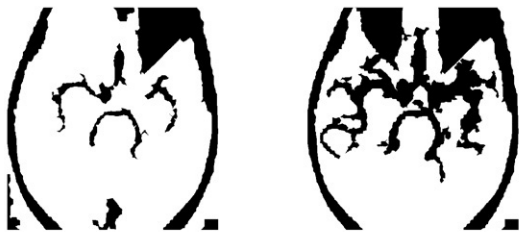

Fig. 12. Several highest-persistence features, cow brain dataset. Left: persistence measured by difference of values of control points for a feature. Right: persistence measured by feature lifetime under anisotropic diffusion (in this case, stable features with infinite lifetime are shown).

The errors are much higher and spread out for sharpening. This is not surprising as sharpening directly increases function value at locations with high-frequency detail, and our algorithm prevents the creation of some of these spurious features.

Fig. 11 shows the numbers of critical points as a function of Laplacian smoothing iteration for several models. Note that the numbers oscillate for Laplacian smoothing but monotonically decrease with topology control. Significant oscillations early in the process of Laplacian smoothing are also present, although not visible because of the large total number of critical points. One can observe that while creation events are clearly a small fraction of the total number of events they have a visible impact on the topological evolution.

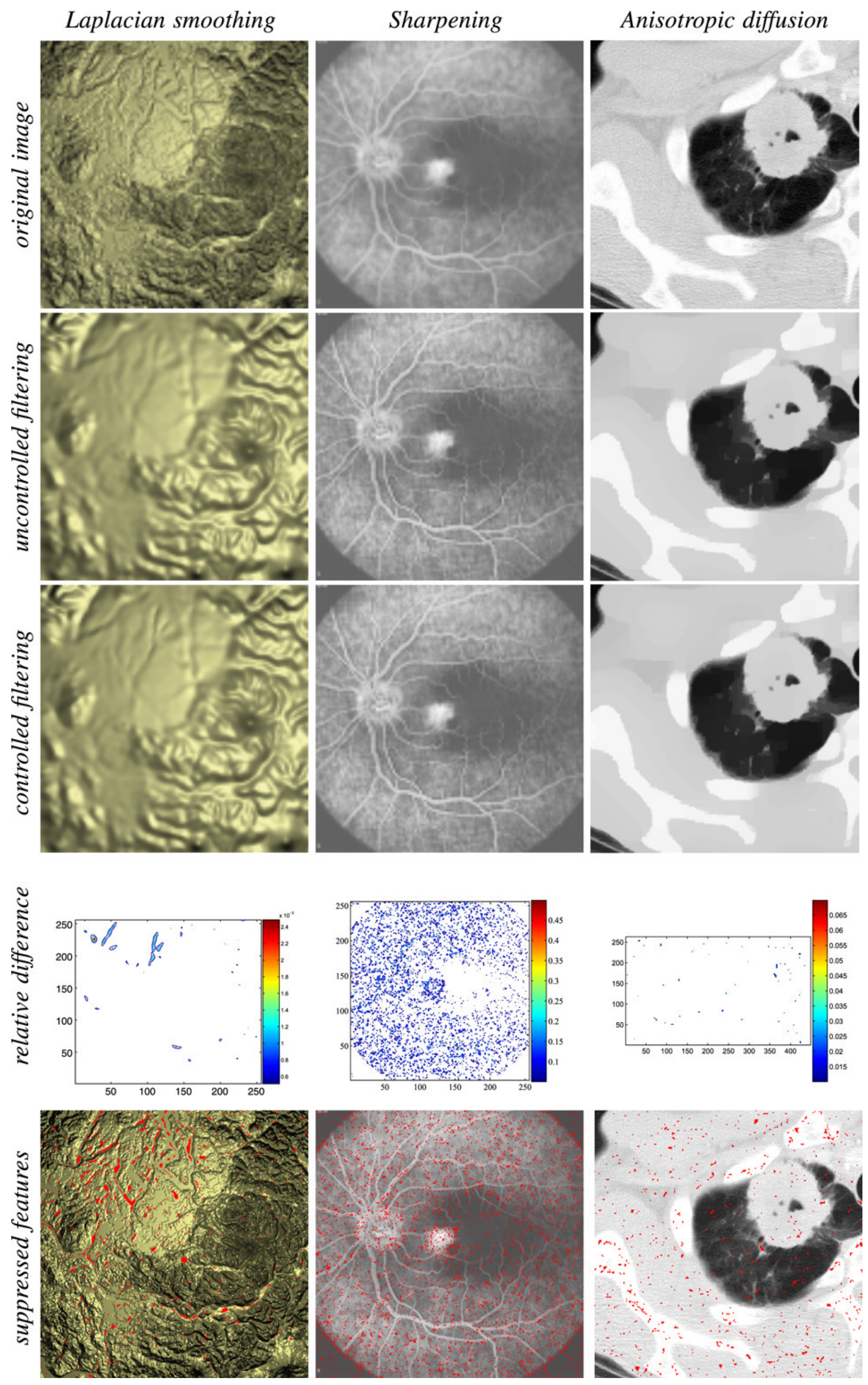

Fig. 13. Filter examples for Laplacian (first column), sharpening (second column) and anisotropic diffusion (third column) filters. The images are: original, filtered without topology control, filtered with topology control, the difference between two filtered images, and the map of new features created without topology control (shown in red). For the anisotropic diffusion only a fragment of the complete image is shown, to make the small-scale features in the original more visible. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 10 illustrates the algorithm performance (256×256 data, Laplacian smoothing, implicit time stepping). One can observe that a significant fraction of the time is spend on the first several steps, due to the need for a large number of inner loop iterations to resolve all topological events, mostly due to small scale noise. Although topology control increased the total time compared to uncontrolled Laplacian smoothing by a factor of 2.22, the first three steps accounted for 43.95% of the difference. Each uncontrolled smoothing step took, on average, 3.10 s; the data set required 128 total smoothing steps. (Results were generated on a 2.4 GHz Xeon.) For later steps, more than one inner loop iteration is rarely needed. If it is acceptable to pre-smooth the image without topology control or otherwise eliminate small-scale features e.g. by setting all values inside the critical contour bounding the feature to the contour value, one can drastically reduce the overhead of iterations in the first few steps. This is true as well for anisotropic diffusion. Sharpening, on the other hand, takes more time per step as virtual time increases. This is because we prevent any topological changes; the sharper the image becomes, the more undesired critical point changes sharpening attempts to affect.

Finally, we observe that by disallowing feature creation events we implicitly obtain a complete or partial topological hierarchy and a measure of feature persistence which can be used for topology visualization and simplification (cf. Carr et al. [19]).

Recall that each topological feature is associated with a pair of critical points (in 2D a saddle and a minimum or maximum). By observing annihilation events, we can establish a feature hierarchy, and use the critical point lifetime as feature persistence. This defines alternative feature persistence measures associated with different filter types. For example, anisotropic diffusion filters would give high persistence to features bounded by well-defined edges. Fig. 12 compares the highest-persistence features given by the simplest persistence definition (value difference at the two critical points defining a feature) with the same number of high-persistence features given by anisotropic diffusion (specifically, stable features with infinite lifetime).

6. Conclusions and future work

We have presented a simple algorithm that ensures that filtering results in topology preservation or monotonic topology simplification in the sense of the reduction of the number of critical points in a scalar field. We have demonstrated that for three filters and a number of test images, the constraints imposed by the topology control algorithm do not significantly affect the filtering process.

Clearly, our algorithm is a first step in this direction. While we have applied it to data sets defined on regular 2D meshes, there are no fundamental limitations on either mesh structure or the dimension of the problem. We plan to explore the behavior of the algorithm in 3D where creation events are more common and more types of topological events may occur. Specifically, critical points have indices 0 through 3; topological events occur between minima and index-1 saddles, index-1 and index-2 saddles, and maxima and index-2 saddles. It is these events that must be detected when a value flips at an edge in a 3D mesh. The structure of the algorithm remains the same. It may, however, be necessary to find ways of improving the algorithm's efficiency.

One significant downside of approaches of this type is that artifacts are hard to predict. While we have observed very few, it would be desirable to have an algorithm which can alter the filter behavior in advance, spreading the modification necessary to prevent disallowed events to a larger number of points and reducing the necessary modification for each point.

Acknowledgements

We would like to thank Chris Wu for his help with algorithm profiling and many useful suggestions; also Chee Yap, Adrian Secord, and other NYU Computer Science colleagues for useful discussions.

References

[1] Bremer P-T, Pascucci V, Edelsbrunner H, Hamann B. A topological hierarchy for functions on triangulated surfaces. IEEE Trans Vis Comput Graph 2004;10:385–96.
[2] He T, Hong L, Varshney A, Wang SW. Controlled topology simplification. IEEE Trans Vis Comput Graph 1996;2(2):171–84.
[3] El-Sana J, Varshney A. Controlled simplification of genus for polygonal models. In: IEEE visualization '97. 1997. p. 403–12.
[4] Guskov I, Wood Z. Topological noise removal. In: Graphics interface 2001. 2001. p. 19–26.
[5] Wood Z, Hoppe H, Desbrun M, Schröder P. Removing excess topology from isosurfaces. ACM Trans on Graph 2004;23(2):190–208.
[6] Milnor J. Morse theory. New Jersey: Princeton Univ. Press; 1963.
[7] Helman JL, Hesselink L. Visualizing vector field topology in fluid flows. IEEE Comput Graph Appl 1991;11(3):36–46.
[8] de Leeuw W, van Liere R. Collapsing flow topology using area metrics. In: VIS '99: Proceedings of the conference on visualization '99. Los Alamitos (CA, USA): IEEE Computer Society Press; 1999. p. 349–54.
[9] Tricoche X, Scheuermann G, Hagen H. A topology simplification method for 2d vector fields. In: IEEE visualization 2000. 2000. p. 359–66.
[10] Tricoche X. Vector and tensor field topology simplification, tracking, and visualization. Ph.D. thesis. Universität Kaiserslautern; 2002.
[11] Tricoche X, Scheuermann G, Hagen H. Continuous topology simplification of planar vector fields. In: IEEE visualization 2001. 2001. p. 159–66.
[12] Freeman H, Morse SP. On searching a contour map for a given terrain profile. J Franklin Inst 1967;248:1–25.
[13] Sircar JK, Cerbrian JA. Application of image processing techniques to the automated labelling of raster figitized contours. In: Int. symp. on spatial data handling. 1986. p. 171–84.
[14] van Kreveld MJ, van Oostrum R, Bajaj CL, Pascucci V, Schikore D. Contour trees and small seed sets for isosurface traversal. In: Symposium on computational geometry. 1997. p. 212–20.
[15] Carr H, Snoeyink J, Axen U. Computing contour trees in all dimensions, In: Symposium on discrete algorithms. 2000. p. 918–26.
[16] Bajaj C, Schikore D. Topology preserving data simplification with error bounds. J Comput Graph 1998;22(1):3–12.
[17] Edelsbrunner H, Harer J, Zomorodian A. Hierarchical Morse complexes for piecewise linear 2-manifolds. In: Proc. 17th Ann. ACM sympos. comput. geom. 2001. p. 70–9.
[18] Edelsbrunner H, Harer J, Natarajan V, Pascucci V. Morse–Smale complexes for piecewise linear 3-manifolds, In: Proc. 19th ann. sympos. comput. geom. 2003. p. 361–70.
[19] Carr H, Snoeyink J, van de Panne M. Simplifying flexible isosurfaces using local geometric measures. In: IEEE visualization 2004. 2004. p. 497–504.
[20] Sohn B, Bajaj C. Time-varying contour topology. IEEE Trans Vis Comput Graphics 2005;12(1):14–25.
[21] Witkin AP. Scale-space filtering. In: International joint conference on artificial intelligence. 1983. p. 1019–22.
[22] Perona P, Malik J. Scale-space and edge detection using anisotropic diffusion. IEEE Trans Pattern Anal Mach Intell 1990;12(7):629–39.
[23] Damon J. Local Morse theory for solutions to the heat equation and Gaussian blurring. J Differential Equations 1995;115(2):368–401.
[24] Florack L, Kuijper A. The topological structure of scale-space images. J Math Imag Vis 2000;12(1):65–79.
[25] Edelsbrunner H, Mücke E. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. ACM Trans Graph 1990; 9(1):66–104.
[26] Demmel JW, Eisenstat SC, Gilbert JR, Li XS, Liu JWH. A supernodal approach to sparse partial pivoting. SIAM J Matrix Anal Appl 1999;20(3): 720–55.