A spatial approach to morphological feature extraction from irregularly sampled scalar fields

Leila De Floriani
Department of Computer Science
University of Genova
deflo@disi.unige.it

Riccardo Fellegara
Department of Computer Science
University of Genova
riccardo.fellegara@unige.it

Federico Iuricich
Department of Computer Science
University of Genova
federico.iuricich@unige.it

Kenneth Weiss
Center for Applied Scientific Computing
Lawrence Livermore National Laboratory
kweiss@llnl.gov

ABSTRACT

Several algorithms have recently been introduced for morphological analysis of scalar fields (terrains, static and dynamic volume data) based on a discrete version of Morse theory. However, despite the applicability of the theory to very general discretized domains, memory constraints have limited its practical usage to scalar fields defined on regular grids, or to relatively small simplicial complexes. We propose an efficient and effective data structure for the extraction of morphological features, such as critical points and their regions of influence, based on the PR-star octree data structure [24], which uses a spatial index over the embedding space of the complex to locally reconstruct the connectivity among its cells.

We demonstrate the effectiveness and scalability of our approach over irregular simplicial meshes in 2D and in 3D with a set of streaming algorithms which extract topological features of the associated scalar field from its locally computed discrete gradient field. Specifically, we extract the critical points of the scalar field, their corresponding regions in the Morse decomposition of the field domain induced by the gradient field, and their connectivity. The spatial index induced by the PR-star octree enables efficient spatial queries on the topological structure of the scalar field, which we demonstrate through window queries on the dataset.

1. INTRODUCTION

The rapid increase in the amount of data produced by sensors and simulations necessitates the creation of efficient and scalable tools for their analysis. Due to their ability to extract essential features from data, topological methods have gained increasing importance in spatial data analysis and scientific visualization. Topological methods are rooted in Morse theory [17], which is the basis for defining decompositions of the domain of a scalar field into regions of influence of the critical points of the field, called *Morse and Morse-Smale complexes*. However, Morse theory applies to smooth functions, while in practical applications we often deal with scalar fields that are regularly or irregularly sampled at discrete locations within a domain. Thus, recent research has focused on combinatorial topological methods which avoid computing derivatives and, thus, are beneficial in the presence of noisy data.

Forman's discrete Morse theory [12] extends Morse theory to cell complexes. Besides its theoretical contributions, the discrete formulation has practical applications in avoiding high-order critical points and in permitting the formulation of robust discrete algorithms [13, 15, 20], which overcome the intrinsic limitations of previous algorithms for approximating Morse complexes [2]. Algorithms based on discrete Morse theory have been developed for regular grids [15, 20] or for triangle and tetrahedral meshes of limited size [15]. A challenging problem when dealing with large simplicial meshes relates to the high memory requirements associated with encoding the irregular connectivity among the cells of the mesh. As such, the need arises for a compact data structure for a simplicial mesh (e.g. a triangle mesh in 2D, a tetrahedral mesh in 3D) which allows localized computations as well as efficient navigation.

Topological data structures for simplicial meshes, which encode the simplices in the mesh in addition to some connectivity relation among them, become too large when the size of the mesh increases. Moreover, when applying topological analysis, we need to store additional information on the critical simplices and on the discrete gradient field. However, as in many mesh processing operations, computing the discrete Morse gradient depends only on the local neighborhood of the mesh vertices. The PR-star octree [24] is an interesting new approach that derives the local connectivity through a spatial index on the mesh geometry. Thus, by trading a reasonable amount of computation at runtime, we can compactly encode all connectivity relations. A benefit of this approach is that the spatial index provides a means of understanding the spatial embedding of the mesh and its associated fields. The PR-star octree can be defined in arbitrary dimensions, but we use it here for triangle meshes embedded in 2D and for tetrahedral meshes embedded in 3D.

Our first contribution is the extension of the algorithm of Robins et al. [20] for discrete Morse gradient computation to simplicial meshes with irregular connectivity. The algorithm in [20] is the only algorithm which extracts the minimal number of critical cells in 2D and in 3D. Although the implementation in [20] is restricted to regular grids, their results are proven for more general cell complexes, and thus are also valid for simplicial meshes. The PR-star octree's compact encoding of topological connectivity provides a scalable tool

The Third ACM SIGSPATIAL International Workshop on GeoStreaming (IWGS) November 5 2012. Redondo Beach, CA, USA

for practical topological analysis of large datasets. Furthermore, it enables us to define optimized local data structures for processing the mesh in each leaf node of the octree, since we only generate the gradient field locally within the domain of the leaf node. This significantly reduces the memory requirements of our algorithm compared to state of the art approaches.

The second contribution of our work is an efficient way of extracting morphological features from the discrete Morse gradient field. The purpose of computing the discrete Morse gradient field is to extract critical points, their region of influence, and the way in which they are connected. Examples of morphological features include: pits with their basins, peaks with their corresponding mountains, separatrices between two mountains or two basins, ridges and valley lines, and saddle connectors [14]. Such features correspond to collections of cells in the Morse complexes, defined by the discrete Morse gradient. Here, we show how to extract morphological features from scalar fields encoded using a PR-star octree. We have developed an algorithm for navigating on a mesh encoded as a PR-star octree through adjacencies. An additional benefit of our approach to feature extraction through the PR-star octree is that it enables *localized* spatial queries on the scalar field. Specifically, we do not need to extract the features of the entire mesh, but we can perform localized queries, for instance, inside a region of interest in the domain, like a box, where the computation cost is proportional to the geometry within the query region.

The remainder of this paper is organized as follows. In Section 2, we discuss some background notions. In Section 3, we discuss related work on spatial indexes and on computation of Morse complexes. Section 4 briefly describes the PR-star quadtree/octree. In Section 5, we describe how to perform the computation of the discrete Morse gradient on the PR-star data structure, while in Section 6, we discuss morphological queries and how they can be answered by performing an efficient navigation in the PR-star octree. Section 7 presents experimental results, while Section 8 draws some concluding remarks and discusses current and future developments.

2. BACKGROUND

Morse theory studies the relationships between the topology of a manifold $\mathbb{M}$ and a function f defined on $\mathbb{M}$ by considering the negative gradient flow of f. For a complete description of the Morse theory, see Milnor [17]. Let f be a C^2 real-valued function (scalar field) defined over a d-dimensional manifold $\mathbb{M}$. A point $p \in \mathbb{M}$ is a *critical point* of f if and only if the gradient of f vanishes at p. Function f is said to be a *Morse function* if all its critical points are non-degenerate (the Hessian matrix $Hess_p f$ of the second derivatives of f at p is non-singular). The number i of the negative eigenvalues of $Hess_p f$ is called the *index* of critical point p. The corresponding eigenvectors point in the directions in which f is decreasing. A critical point of index i, $0 \leq i \leq d$, p is called an *i-saddle*, where a 0-saddle is also referred to as a *minimum* and a d-saddle as a *maximum*. If $d = 2$, there are three types of critical points, namely minima, saddles and maxima. If $d = 3$, there are four types of critical points, namely minima, 1-saddles, 2-saddles and maxima.

An *integral line* of a function f is a maximal path that is everywhere tangent to its gradient ∇f. It follows the direction in which the function has the maximum increasing growth. Two integral lines are either disjoint or identical. Each integral line starts at a critical point of f, called its *origin*, and ends at another critical point, called its *destination*. Integral lines that converge to a critical point p of index i cover an i-cell called the *descending manifold* of p. Dually, integral lines that originate at p cover an $(n-i)$-cell called the *ascending manifold of p*. The descending manifolds decompose $\mathbb{M}$ into a cell complex, called the *descending Morse complex* of f on $\mathbb{M}$. Dually, the ascending manifolds form the *ascending Morse complex* of f on $\mathbb{M}$. A Morse function f is called a *Morse-Smale function* if and only if each non-empty intersection of a descending and an ascending manifold is transversal. This means that each connected component of the intersection (if it exists) of the descending i-cell of a critical point p of index i, and the ascending $(n-j)$-cell of a critical point q of index j, $i \geq j$, is an $(i-j)$-cell. The connected components of the intersection of descending and ascending cells of a Morse-Smale function f decompose $\mathbb{M}$ into a *Morse-Smale complex*. If f is a Morse-Smale function, then there is no integral line connecting two different critical points of f of the same index.

Forman's theory [12] is an elegant adaptation of classical Morse theory to functions defined over a cell complex. The main purpose of Forman's theory is to develop a discrete setting in which almost all the main results from Morse theory are valid. This goal is achieved by considering a function F defined on all cells, and not only on the vertices, of a cell complex. Here, we define it for a simplicial mesh Σ, i.e., for a cell complex in which all the cells are simplices. Intuitively, a function F is a discrete Morse function if for any p-simplex σ, all the $(p-1)$-simplices on its boundary have a lower F value than σ, and all the $(p+1)$-simplices in its co-boundary have a higher F value, with at most one exception. If there is such an exception, it defines a pairing of cells of Σ, called a *discrete* (or *Forman*) *gradient vector field* V. Otherwise, p-simplex σ is a *critical* simplex of index i.

More formally, a function $F : \Sigma \to \mathbb{R}$ is a *discrete Morse function* if, for every p-simplex σ, the following conditions are satisfied:

$$\#\left\{\tau^{p+1} > \sigma : F(\tau) \leq F(\sigma)\right\} \leq 1$$

and

$$\#\left\{\nu^{p-1} < \sigma : F(\nu) \geq F(\sigma)\right\} \leq 1.$$

These inequalities cannot be equalities at the same time [12]. This means that, for a p-simplex σ, we cannot find simultaneously a $(p-1)$-dimensional face ν of σ and a $(p+1)$-dimensional co-face τ of σ, such that $F(\tau) \leq F(\sigma) \leq F(\nu)$.

A p-simplex $\sigma \in \Sigma$ is a *critical simplex* of index p if the following condition is satisfied:

$$\#\left\{\tau^{p+1} > \sigma : F(\tau) \leq F(\sigma)\right\} = \#\left\{\nu^{p-1} < \sigma : F(\nu) \geq F(\sigma)\right\} = 0.$$

This implies that the absolute minimum of a discrete Morse function F on a simplicial complex Σ occurs at a vertex, while if Σ is a triangulation of a closed d-manifold, then the absolute maximum of F occurs at a d-simplex.

As noted by Forman [12], it is not easy to construct discrete Morse functions; it is simpler to define a *discrete vector field*. Intuitively, a discrete vector field can be viewed as a collection of *arrows*, connecting a p-simplex of Σ to an incident $(p+1)$-simplex of Σ, such that each simplex is a *head* or a *tail* of at most one arrow. Critical simplices are those simplices that are neither the head nor the tail of any arrow. A discrete vector field V is a *Morse gradient vector field* if there are no closed V-paths in V. A *V-path* is a sequence $\sigma_0, \tau_0, \sigma_1, \tau_1, ..., \sigma_{r+1}$ of p-simplices σ_i and $(p+1)$-simplices τ_j, $i = 0, ..., r+1$, $j = 0, ..., r$, such that $(\sigma_i, \tau_i) \in V$, $\tau_i > \sigma_{i+1}$, and $\sigma_i \neq \sigma_{i+1}$.

There is a correspondence between discrete Morse functions and discrete gradient vector fields. Namely, for each discrete Morse function F, a discrete gradient vector field V can be constructed. A discrete gradient vector field V of F is a vector field which is obtained from F by noticing that non-critical simplices come in pairs, and by drawing an arrow from a p-simplex σ to a $(p+1)$-simplex τ if $\tau > \sigma$ and $F(\tau) \leq F(\sigma)$. If F is a discrete Morse function, then each simplex of Σ is the head or the tail of at most one arrow, and the critical simplices are those simplices that are neither the head nor the tail of any arrow.

3. RELATED WORK

A variety of hierarchical spatial indexes have been proposed in the literature for points, polygonal maps, boundary representations of objects, triangle and tetrahedral mesh [21]. Hierarchical spatial indexes for points in the Euclidean space, such as PR quadtrees/octrees and PR kd-trees, contain the points only in their leaf nodes and, thus, the shape of the tree is independent of the order in which the points are inserted. The PMR quadtree [19] is a spatial index for a collection of edges in the plane (not necessarily forming a polygonal map). The family of *PM quadtrees* [22] extend the PR quadtree to represent polygonal maps considered as structured collections of edges, and differ in their refinement rules. In [7] a hierarchical spatial index for triangle meshes is proposed which extends a PM-quadtree. The triangles (instead of the edges) in the mesh are indexed by the leaf nodes of the quadtree. PM quadtrees have also been extended to index the boundary of a polyhedral object in space [4, 18]. The subdivision rules are similar to those of PM-quadtrees but the faces of the object are considered instead of the edges of the mesh. In [8], a collection of spatial indexes for tetrahedral meshes, called *tetrahedral trees* are introduced. One of the tetrahedral trees extends the PMR quadtree by indexing the tetrahedra in the mesh instead of the edges of the map, while another tetrahedral tree is a direct extension of the PM octree to tetrahedral meshes.

In [24], a fundamentally and conceptually different data structure for tetrahedral meshes, called the *PR-star octree*, has been presented, in which the hierarchy is not a spatial index on the mesh (i.e. to support efficient spatial queries such as point location), but, rather, is a tool to support efficient retrieval of topological connectivity (i.e. for topological queries), thus encoding a minimum topology and trading spatial relations for topological ones.

Other approaches propose localized computations by using a spatial index to reduce memory requirements for out-of-core [5] or memory intensive mesh processing [9], or by developing a reduced data structure for a simplicial meshes [3]. Cignoni et al. [5] introduce an external memory spatial data structure for processing large triangle meshes to support compact out-of-core processing of large triangle meshes. Dey et al. [9] use an octree to index a large triangle mesh for localized Delaunay remeshing. In [3], a very light data structure for computing a Delaunay simplicial mesh in medium dimensions (up to 6 dimensions) is proposed which encodes only the edges of the mesh globally and reconstructs the connectivity only locally when required by the computation.

There have been basically two approaches in the literature to extend the results of Morse theory and represent Morse and Morse-Smale complexes in the discrete case. One approach is discrete Morse theory [12], that we have reviewed in Section 2. The other approach, introduced in [11] in 2D, and in [10] in 3D, is based on Banchoff's extension of Morse theory to piecewise-linear manifolds and functions [1]. It provides a combinatorial description of the Morse-Smale complex of a function f defined at the vertices of a simplicial mesh. Surveys of algorithms for computing Morse and Morse-Smale complexes in 2D and 3D based on this theory and based on a watershed approach can be found in [2, 6].

Recently, there has been a lot of attention on algorithms rooted in Forman's discrete Morse theory since they are generally more efficient from a computational point of view than the previous approaches, which try to simulate the behavior of a continuous function. The problem of computing an optimal Forman gradient vector field V on a 2D simplicial mesh has been discussed in [16]. King et al. [15] present an algorithm to compute a Forman gradient vector field V on a scalar field f defined on the vertices of a 3D simplicial mesh. The same algorithm can be applied to 2D simplicial meshes embedded in 3D. In [13], Gyulassy et al. use Forman theory with a divide-and-conquer technique to compute an approximation of the Morse-Smale complex of a scalar field f defined on the vertices of a regular 3D grid. It is not guaranteed that the constructed discrete gradient vector field V, if applied to a triangulated domain, points in the direction in which the scalar field f is decreasing. In [20], Robins et al. define an algorithm to compute the Forman discrete vector field on 2D and 3D regular grids, and they show that the result produced is optimal in the sense that the discrete gradient vector field has a minimal number of critical cells.

4. PR-STAR QUADTREES AND OCTREES

In this section, we describe the *PR-star* data structures [24]: PR-star quadtrees for triangle meshes in 2D, and PR-star octrees for tetrahedral meshes in 3D. In contrast to topological data structures, which explicitly encode the connectivity among mesh elements, or to spatial data structures, which index the elements for efficient spatial queries, PR-star quadtrees and octrees use the spatial index induced by a quadtree (in 2D) or an octree (in 3D) to efficiently generate local application-dependent topological data structures at runtime. In [24], algorithms have been presented to extract some topological relations on the simplices based on the PR-star tree, but it has not been shown how to efficiently navigate in a mesh through adjacencies when using a PR-star tree.

The PR-star tree is based on the *Point Region quadtree (PR quadtree)* [21], which is a spatial index on a set of points P in a d-dimensional domain. The domain decomposition is controlled by a single parameter k_v which determines the maximum number of points indexed by a leaf node. The insertion of a new point into a *full* leaf in the tree causes the leaf to split and its indexed points to be redistributed among its children. Thus, the domain decomposition of a PR quadtree is independent of the insertion order of its points.

The *PR-star quadtree/octree* for a triangle/tetrahedral mesh Σ consists of (a) an array P of Σ's vertices, which encode the geometry of the mesh; (b) an array T of triangles (in 2D) or tetrahedra (in 3D). Each element in $T$is encoded in terms of the indices of its three/four vertices within P; (c) an augmented PR quadtree/octree N, whose leaf nodes index a subset of vertices from P, as well as all the elements from T that are incident in these vertices.

We use a more compact representation for the leaves of the PR quadtree/octree compared with [24], where we exploit the spatial locality provided by the tree through a reindexing of arrays P and T. Besides the hierarchical information associated with the tree (e.g. pointers to the parent node and to the set of children nodes), each leaf node n encodes: the range of the indices v_{start} and v_{end} in P of the vertices in n; the range of the indices t_{start} and t_{end} in T of

the triangles (tetrahedra) that are completely contained in n; and a pointer to a list of the remaining elements from T that are incident in these vertices i.e. they have at least one vertex outside the domain of n.

The basic paradigm for performing operations on a mesh encoded as a PR-star quad tree/octree is to locally process the mesh in a streaming manner by iterating through the leaf nodes of the tree. For each leaf node n, a local application-dependent data structure is built, which is then used to process the local geometry. After we finish processing node n, we discard the local data structure and move on to the next node.

The basic ingredients in any data structure which encodes a mesh is the representation of a suitable subset of the topological relations among the simplices. For example, to build the local *Vertex-Triangle* relation (*Vertex-Tetrahedron* relation) for the vertices P_n in a node n, i.e., the set of triangles (tetrahedra) incident in each vertex v of P_n), the algorithm iterates through the vertices of the triangles (tetrahedra) T_n in n. For each vertex v of a triangle (tetrahedron) t indexed by n, we add the index of t in T to the list of triangles (tetrahedra) incident in v. Since the indexed vertices are in contiguous positions in the global vertex array, and there are at most k_v vertices associated with leaf node n of N, the local data structure is an array of size k_v. Each position in this array corresponds to a vertex indexed by n and points to an (initially empty) list of indices from T. For simplicity, in the remainder of the paper, we will always use the term PR-star octree to indicate a quadtree or an octree.

5. COMPUTATION OF THE DISCRETE MORSE GRADIENT FIELD

In this section, we adapt the algorithm of Robins et al. [20] for extracting the discrete Morse gradient field to simplicial meshes. The algorithm by Robins et al. is defined for cubical cell complexes and has been applied to homology computation of 2D and 3D images. The optimality results (on the minimal number of critical cells) proven in [20] hold also for cell complexes, and thus for simplicial meshes. In our work, we have implemented two versions of the algorithm for simplicial meshes; one using a compact topological data structure to encode the mesh and the other using the PR-star octree, described in Section 4. For the former, we use the *Indexed data structure with Adjacencies (IA data structure)*, which encodes only the vertices and the top simplices of the mesh (triangles in 2D, tetrahedra in 3D), and the adjacencies of the d-simplices along their common $(d-1)$-simplices (edges in 2D, triangles in 3D).

The discrete Morse computation algorithm takes as input a simplicial mesh and the field values given at the vertices of the mesh and outputs a list C of the critical simplices as well as the discrete gradient field encoded as a collection of *arrows* from an i-dimensional simplex to its paired $(i+1)$-dimensional simplex. This is generated via homotopic expansions of the *lower star* of each vertex of the input mesh. The *lower star* of a vertex v consists of the simplices σ incident in v with $f_{max}(\sigma) = f(v)$ and $f_{max}(\sigma) = \max_{p \in \sigma} f(p)$. Since pairing occurs only between simplices in the same lower star, each lower star can be treated independently. Figure 1 illustrates a discrete gradient vector field computed on a simplified terrain dataset.

From a data structure point of view, the algorithm requires an efficient computation of the star of a vertex v, i.e., an efficient way to extract the set of simplices incident at v. Our implementation based on the PR-star octree exploits the capability of the PR-star octree in efficiently extracting the simplices incident in a vertex, as discussed in Section 4.

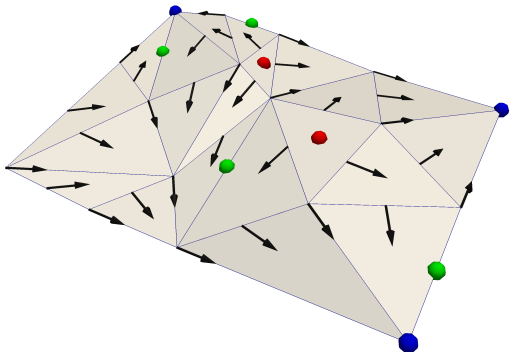


Figure 1: Example of a discrete gradient vector field on a triangulated terrain. Paired simplices are marked with arrows from the *origin* to its *destination*. Critical simplices are marked with blue (minima), green (saddle) and red (maxima) dots.

When the scalar field is defined over a cubical cell complex (regular grid) [20, 14], arrows defining the gradient vector can be efficiently encoded using a grid with twice the resolution along each direction. We now describe our data structure for encoding the vector field map for scalar fields defined on simplicial meshes. A vector pair (σ, τ) is a mapping from an i-simplex σ to an $(i+1)$-simplex τ, which we denote as $V[\sigma] = \tau$. In both our representations (IA data structure and PR-star octree), we have an efficient index on the vertices P and on the top simplices T (i.e. the triangles in 2D or the tetrahedra in 3D), but not on the faces of the top simplices (i.e. the edges in 2D and 3D, and the triangular faces in 3D). Thus, we encode each map of vector pairs differently, depending on the dimension of the mesh and of the mapped elements.

In 2D, there are two types of maps, V_{01}, from vertices to edges, and V_{12}, from edges to triangular faces. $V_{01} : v \to e$ is encoded as a map from the index in P of v to the index in P of the other endpoint of e, while $V_{12} : e \to f$ is encoded as a map from a pair of indices in P of the endpoints of e to the index in T of f, requiring, respectively, two, and three references per gradient pair.

In 3D, there is an additional map V_{23}, from faces to tetrahedra. $V_{01} : v \to e$ is encoded as a map from the index in P of v to the index in P of the other endpoint of e, $V_{12} : e \to f$ is encoded as a map from a pair of indices in P of the endpoints of e to the index in P of the vertex of f not incident in e, and $V_{23} : f \to t$ is encoded as a map from a triple of indices in P of the endpoints of e to the index in T of t, requiring, respectively, 2, 3 and 4 references per arrow.

Since the vector field computed by the algorithm is a discrete Morse gradient vector field, each i-simplex of Σ is paired with only one $(i \pm 1)$-simplex encoded in one of the maps, or it is not paired to any other simplex and stored in the set of critical simplices. The critical simplices are encoded in different sets based on their dimension. Each simplex is represented, similarly to the representation into the gradient vector, through one index if it is a vertex or a top simplex (triangle in 2D or tetrahedron in 3D), or by using two indices into P for edges and three indexes for triangles (in 3D).

6. LOCALIZED FEATURE EXTRACTION

In this section, we discuss how we use the local discrete gradient field from Section 5 to extract morphological features from 2D and 3D scalar fields indexed by a PR-star octree.

Morphological features are used to analyze the topology of the scalar field. As such, these features are typically understood collectively either by analyzing all features of a given dimension, (e.g. the 1-skeleton of the Morse complex, or of the Morse-Smale complex) or by combining the features within a local region (e.g. through a window query). Examples of morphological features are pits with their basins, peaks with their corresponding mountains, separatrices between two mountains or two basins, ridges and valley lines, saddle connectors. Such features correspond to collection of cells in the Morse complexes, defined by the discrete Morse gradient vector field. A complete set of morphological features for 3D scalar fields is presented in [14].

We first describe the general algorithm for extracting a single feature, i.e. the region within the Morse complex corresponding to a single critical point of the field. We then describe our streaming algorithm for extracting all features of a given dimension through a traversal of the PR-star octree leaf nodes. Finally, we describe an adaptation that illustrates the power of our spatial approach. Namely, through a simple modification, we can efficiently support window queries on the topological features, by extracting the region of influence of all the critical points of a certain kind within a specified window on the field domain (a rectangle for 2D scalar fields, a box in the 3D case).

6.1 Extracting morphological features

The morphological features of a scalar field are defined by the ascending and descending manifolds of its critical points. For example, for a 2D scalar field, the descending 2-manifolds define the regions of influences of the maxima, the descending 1-manifolds the regions of influence of the saddle points and their connection to the minima. For a 3D scalar field, the descending 3-manifolds define the regions of influences of the maxima, the descending 2-manifolds the regions of influence of the 2-saddles, the descending 1-manifolds the regions of influence of the 1-saddles, and their connections to the minima. The reverse holds for the ascending manifolds. For instance, the ascending 2-manifold of a minimum in a 2D scalar field is the basin associated with it. The i-dimensional features are then expressed as collections of i-simplices from the underlying mesh that cover a descending or ascending i-manifold.

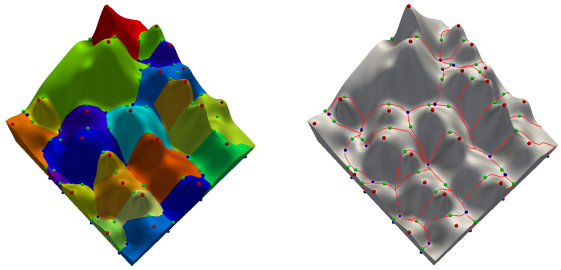
Generally an ascending or a descending i-manifold feature is extracted by traversing the arrows of the discrete Morse field V and by collecting all the i-simplices in the ascending/descending i-manifold. For example, for a 2D scalar field discretized as a triangle mesh, extracting a descending 2-manifold, which corresponds to a maximum (i.e., a triangle f in the mesh) corresponds to collecting all the triangles belonging to the 2-manifold obtained by navigating the vector field starting from f (see Figure 2a). Similarly, extracting the 1-manifold corresponding to a saddle (i.e., an edge e in the mesh) corresponds to a collection of edges belonging to the 1-manifold obtained by navigating V starting with e (see Figure 2b).

The extraction algorithm can be defined in a dimension independent manner. We focus here, for brevity, on the extraction of descending i-manifolds, which are the region of influence of i-saddles. Recall that 0-saddles are minima and d-saddles are maxima, where d is the dimension of the scalar field (and thus of the Morse complexes). Let β be a critical i-simplex. The extraction of its descending i-manifold

Algorithm 1 EXTRACTDESCENDINGMANIFOLD(β)

Require: β is a critical i-simplex, $0 < i \leq d$
Require: Q is a queue of $(i-1)$-simplices
Require: Σ_i is a simplicial i-complex containing the i-simplices in the extracted feature i-manifold

```
// Initialize queue
for all facets σ of β do
  Q.enqueue(σ)
// Iterate through (i−1)-simplices in queue
while Q is not empty do
  σ ← Q.pop()
  if ISTAIL(σ) then
    τ ← HEAD(σ)
    Σ_i.add(τ)
    for all facets σ' of τ, where σ' ≠ σ do
      Q.enqueue(σ')
return Σ_i
```



(a) (b)

Figure 2: Example of features extracted from a synthetic data set formed by a set of Gaussian surfaces intersecting a plane. The descending 2-manifolds are shown in (a) and the descending 1-manifolds in (b). Blue, green and red dots correspond to minima, saddles and maxima, respectively.

is described in Algorithm 1, where ISTAIL(σ) is a predicate that returns TRUE if $(i-1)$-simplex σ is a tail of some gradient pair, and FALSE otherwise. HEAD(σ) returns the head of the gradient arrow (i.e. an i-simplex) in which σ is the tail. Since there are no cycles in the gradient field, the algorithm is guaranteed to terminate and the set of visited i-simplices forms the i-manifold corresponding to critical i-simplex β.

The input to the algorithm is a critical simplex β and a discrete Morse gradient field V. The output is a simplicial i-complex covering an i-cell of the descending/ascending Morse complex (i.e., an i-manifold). To extract features of the ascending Morse complex we have to invert the gradient navigation.

6.2 Streaming feature extraction

In this section, we describe our streaming feature extraction algorithm, which extracts the entire set of i-dimensional features from the mesh Σ endowed with a scalar field and indexed by a PR-star octree N. Our algorithm extracts all such features in a single pass through the octree leaf nodes. Upon completion of the algorithm, we generate a simplicial i-complex consisting of the simplices covering the i-dimensional feature cells.

In contrast to the description above, which assumes the existence

of a global gradient vector field, when processing an octree node n, we only generate a local vector gradient field V_{LOCAL} involving the simplices in the lower star of the indexed vertices P_n. Thus, we must encode auxiliary data structures to maintain a subset of the gradient vector field generated by previous octree nodes. Furthermore, since not all paths can be fully extracted locally, we defer the completion of these features until we can visit such nodes. This is accomplished by saving the terminal explored simplex on these *dangling paths*.

By analyzing Algorithm 1, we see that each arrow in the gradient field is visited at most once, i.e. by the extraction traversal initialized by its critical simplex. Thus, our auxiliary data structure maintains the set of unvisited arrows from the global gradient field that point to i-simplices, which we denote as V_{GLOBAL}. When a gradient arrow from V_{GLOBAL} is visited during the path traversal started from another octree leaf node, it is removed from the map.

The second auxiliary data structure Σ_{DANGLE} is used to continue paths begun while processing previous octree nodes. and encodes the terminal simplices visited along *dangling paths*. Specifically, we encode this data structure as a map from the index of the leaf node in which the interrupted path will continue to the set of i simplices terminating the dangling path.

Our local algorithm for extracting i-dimensional features from each octree leaf node n performs four steps:

1. *Gradient vector computation.* We begin by generating the local gradient vector field V_{LOCAL} for leaf octree node n. This consists of the arrows defined by the lower stars of P_n, the vertices indexed by n.
2. *Dangling path expansion.* In order to minimize the global memory usage, we first try to complete the paths initiated by previous octree nodes. For each i-simplex σ in $\Sigma_{\text{DANGLE}}[n]$, whose gradient is stored in V_{GLOBAL}, we continue the gradient path within n. After processing each arrow from the gradient vector field (V_{LOCAL} or V_{GLOBAL}), we remove it from its associated gradient field.
3. *New path expansion.* Once all the dangling paths have been expanded, we begin extracting the features associated with the critical i-simplices indexed by n. This traversal uses (and then discards) the arrows from both V_{LOCAL} or V_{GLOBAL}, as appropriate. If an expanding path at i-simplex σ reaches the boundary of n and continues into unvisited octree node n', we add σ to $\Sigma_{\text{DANGLE}}[n']$.
4. *Gradient vector conversion.* Finally, after all the paths inside the leaf node have been completed (because they reached the destination critical simplices or the boundary of the leaf node) the unvisited gradient pairs from V_{LOCAL} that terminate in an i-simplex are added to V_{GLOBAL}. Note that only the *<(i-1)-simplex, (i)-simplex>* gradient pairs have to be saved.

The output produced by our feature extraction varies depending on the feature dimension. Since the top simplices already provides us with a globally consistent mapping, the output of the d-manifold extraction for the descending Morse complex is an array of size $|T|$ of labels, where the label of each d-simplex is the index of its associated critical d-simplex. For the remaining i-manifolds, the output of our extraction is a non-manifold simplicial i-complex in which each i-simplex is encoded as a tuple of $i+1$ vertices and a label indicating its corresponding critical point.

We note that, while the above algorithm describes the extraction of a single set of features (e.g. the i-dimensional features), we can easily modify the algorithm to concurrently extract multiple feature sets, thereby leveraging the computationally expensive generation of the gradient vector field. For example, we can simultaneously extract the d sets of features encompassing the entire Morse complex.

7. EXPERIMENTAL RESULTS

We have compared two implementations of the gradient computation and of the morphological feature extraction, one using the *Indexed data structure with Adjacencies (IA data structure)* and the other using the PR-star octree, and we have validated our approach across two type of extractions: one based on global feature extraction and one based on a windowed feature extraction inside a region of interest. We have performed experiments on fourteen tetrahedral meshes whose sized range from 1.5 million to to 35 million tetrahedra, and on three triangular meshes whose sizes range from between 4 million to 19.5 million triangles. The hardware configuration used is an Intel i7-2600K CPU at 3.20Ghz with 16 GB of RAM.

First, we have extracted the discrete Morse gradient vector field, and we have evaluated both the storage costs of the underlying data structures (IA and PR-octree) as well as the storage costs and time requirements for computing the gradient on each of them, as shown in Table 1. The first column of Table 1 shows the value of the bucket threshold k_v used for the PR-octree. Changing the value of k_v for the PR-star octree does not significantly influence the storage requirements for feature extraction, and, thus, we only present the results for a single value of k_v. For the sake of brevity, we report on a representative subset of the results obtained, we describe the results for two highly irregular tetrahedral meshes (FIGHTER2 and F16_DENSITY), three large tetrahedral meshes with semi-regular connectivity extracted from a regular grid using regular simplex bisection [23] (BONSAI, VISMALE and FOOT), and three terrain datasets (MAUI, BAIA and PUGET). Our experiments show (column MESH in Table 1) that the PR-star requires approximately 60% the space of the IA for encoding the mesh.

The gradient generation algorithm for the PR-star is entirely local, and uses constant storage with respect to the IA data structure (i.e, a fraction of one percent). We can see the efficiency of the algorithm from the reduced generation times achieved by the PR-star data structure. Despite having to reconstruct the (lower) star of every vertex, we achieve a saving between 10% to 30% since we do not store the global gradient field. We achieved slightly better timing performance for lower values of k_v (not reported in Table 1).

In the rightmost columns of Table 1 we show the storage costs and the timings for each independent feature extraction. We show an example of global feature extraction for a triangulated terrain in Figure 3.

Global storage costs are lowest for the PR-star when extracting d-manifold features, (3-manifolds in 3D and 2-manifolds in 2D), since all entries in the global structures are guaranteed to be visited and removed at some point during the navigation. For 3-manifold extraction the PR-star uses between 1% to 30% of the storage needed by the IA, while for 2-manifolds, it uses approximately 30% of the IA's storage for tetrahedral meshes, and between 15% and 30% for triangle meshes. For 1-manifold extraction, it uses only 6% of the IA's storage for tetrahedral meshes and around 25% for triangle meshes. If we consider the entire storage required by the data structures for encoding the mesh and by the auxiliary representations used during computation of the gradient and for the extraction of

the feature, PR-octree implementation uses from 30% to 40% less memory than an IA-based approach.

In terms of extraction times for the sets of *i-manifolds*, the PR-star requires from 3% to 35% less time for 3-manifolds. For 2-manifolds extraction, we obtain similar timings on the smaller meshes in our dataset, while for larger meshes, the PR-star is faster, requiring at most 30% less time. For 1-manifold extraction, we found the PR-star to be slower than the IA. We note that this step is generally quite fast (on the order of hundredths to tenths of a second), and is several orders of magnitude faster than gradient extraction, which must be performed first. When we compare the relative performances across the whole feature extraction process (gradient computation plus feature extraction), the PR-star is 5% to 30% faster than the IA.

The above streaming algorithm can be modified to enable efficient windowed feature extraction queries on the field morphology, in which the gradient vector only needs to be extracted within a local neighborhood, such as an axis-aligned box. Specifically, we extract the gradient vector field for all the nodes, in the PR-star octree, that intersect the window, and extract the *i*-dimensional features for all critical simplices within the window. In this query, the PR-star implementation is clearly superior to the IA one since it only needs to compute a subset of the gradient vector, while the IA implementation has to compute the entire gradient vector, even if a small part is needed to execute the query. Our experiments (not reported here for brevity) show that the PR-star octree uses on average from 80% to 99% less memory than the IA-based implementation, depending on the window size.

8. CONCLUDING REMARKS

We have introduced an efficient tool based on the PR-star octree for extracting morphological features from scalar fields defined on irregular simplicial meshes in 2D and 3D, and for computing its Morse gradient field in a streaming manner.

We have compared the gradient computation and feature extraction algorithms to a compact state of the art topological data structure, the IA data structure. Our experiments indicate that the PR-star implementation is significantly more efficient in terms of storage and computation with respect to the IA data structure when extracting morphological features on the whole domain, and it completely outperforms the latter when extracting such features on a portion of the domain. One of the major limitations of existing methods for morphological analysis is due to the lack of memory-efficient implementations. This reduces their feasibility for real-world applications. We believe that the combination of spatial indexing with scalar field analysis can lead to many fruitful discoveries in the field.

In our current and future work, we plan to explore several directions. We plan to perform a parallel implementation of the gradient computation algorithm, since the processing of the lower stars of the vertices can be done independently.

The output of our queries is a collection of *i*-manifolds, each expressed as a 'soup of simplices' of the appropriate dimension. It would be useful to also generate the connectivity of the extracted ascending or descending Morse complex. Since the Morse complex is spatially embedded, we would like to index this extracted complex with an augmented PR-star data structure. This will enable us efficiently process the extracted mesh, e.g. to perform topological cancellations on the complex for efficient noise removal.

Acknowledgements

This work has been partially supported by the Italian Ministry of Education and Research under the PRIN 2009 program, and by the National Science Foundation under grant number IIS-1116747. It has also been performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-CONF-563559). Volumetric datasets are courtesy of the Volvis repository (F16_DENSITY, BONSAI, FOOT), C. Silva (FIGHTER_2), and the Volume Library (volib) (BUCKY, VISMALE). Terrain datasets are courtesy of the Virtual Terrain Project (VTP) (BAIA and MAUI), the Geometric Models Archive (PUGET).

9. REFERENCES

[1] T. Banchoff. Critical Points and Curvature for Embedded Polyhedral Surfaces. *American Mathematical Monthly*, 77(5):475–485, 1970.

[2] S. Biasotti, L. De Floriani, B. Falcidieno, P. Frosini, D. Giorgi, C. Landi, L. Papaleo, and M. Spagnuolo. Describing shapes by geometrical-topological properties of real functions. *ACM Computing Surveys*, 40(4):12:1–12:87, October 2008.

[3] J.-D. Boissonnat, O. Devillers, and S. Hornus. Incremental construction of the Delaunay triangulation and the Delaunay graph in medium dimension. In *Proceedings Symposium on Computational Geometry*, SCG '09, pages 208–216, New York, NY, USA, 2009. ACM.

[4] I. Carlbom, I. Chakravarty, and D. Vanderschel. A hierarchical data structure for representing the spatial decomposition of 3D objects. *IEEE Computer Graphics and Applications*, 5(4):24–31, 1985.

[5] P. Cignoni, C. Montani, C. Rocchini, and R. Scopigno. External memory management and simplification of huge meshes. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):525–537, 2003.

[6] L. Čomić, L. De Floriani, and F. Iuricich. Modeling Three-Dimensional Morse and Morse-Smale Complexes. *Dagstuhl Seminar 11142 - Innovations for Shape Analysis: Models and Algorithms*, To appear.

[7] L. De Floriani, M. Facinoli, P. Magillo, and B. Dimitri. A hierarchical spatial index for triangulated surfaces. In *Int. Conf. on Computer Graphics Theory and Applications (GRAPP)*, pages 86–91, 2008.

[8] L. De Floriani, R. Fellegara, and P. Magillo. Spatial indexing on tetrahedral meshes. In *Proceedings ACM SIGSPATIAL GIS*, GIS '10, pages 506–509. ACM, 2010.

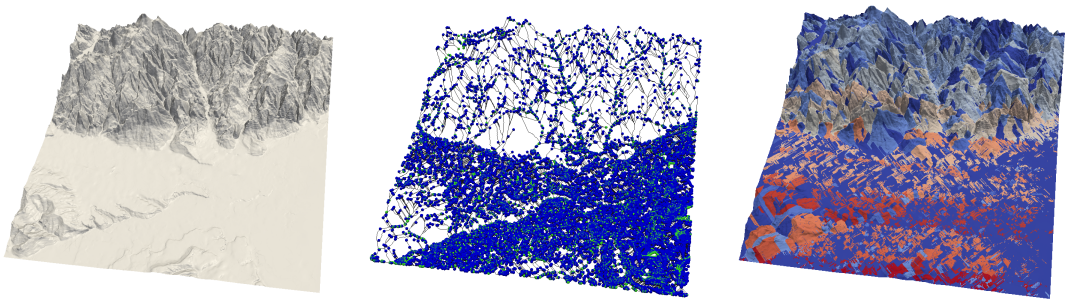
[9] T. Dey, J. Levine, and A. Slatton. Localized Delaunay refinement for sampling and meshing. *Computer Graphics Forum*, 29(5):1723–1732, 2010.

[10] H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Morse-Smale Complexes for Piecewise Linear 3-Manifolds. In *Proceedings 19th ACM Symposium on Computational Geometry*, pages 361–370, 2003.

[11] H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical Morse Complexes for Piecewise Linear 2-Manifolds. In *Proceedings 17th ACM Symposium on Computational Geometry*, pages 70–79, 2001.

[12] R. Forman. Morse Theory for Cell Complexes. *Advances in Mathematics*, 134:90–145, 1998.

[13] A. Gyulassy, P. T. Bremer, B. Hamann, and V. Pascucci. A practical approach to Morse-Smale complex computation: Scalability and generality. *IEEE Trans. Vis. Comput. Graph.*,



(a) (b) (c)

Figure 3: Global feature extraction on BAIA. (a) Original dataset. (b) Extracted 1-manifolds with minima (blue) and saddles (green). (c) Extracted 2-manifolds, colors indicate distinct regions.

Table 1: Storage costs (expressed as MBs) and timings for the implementations based on the PR-star octree and the IA data structures. Column $|T|$ shows the number of top simplices of each data set. Columns *mesh* show the storage cost for the data structures encoding the mesh (IA data structure and PR-star octree). In columns %, we indicate the percentage of memory or time used by the PR-star with respect to the IA. The top five meshes tetrahedral meshes, the bottom three are triangle meshes.

data set	$\|T\|$	k_v	mesh		gradient vector				1-manifold				2-manifold				3-manifold			
			storage		timings		storage		timings		storage		timings		storage		timings		storage	
			tot	%	tot	%	tot	%	tot	%	tot	%	tot	%	tot	%	tot	%	tot	%
FIGHTER2	1.40M	IA	48	–	28.62	–	111	–	0.02	–	111	–	0.46	–	111	–	1.03	–	111	–
		400	30	**62**	19.45	**68**	0.20	**0.18**	0.04	**188**	7.4	**6.6**	0.45	**97**	40	**36**	0.77	**75**	36	**32**
F16_DENSITY	6.35M	IA	215	–	131.40	–	500	–	0.06	–	500	–	3.82	–	500	–	9.13	–	500	–
		800	131	**61**	95.55	**73**	0.41	**0.08**	0.40	**666**	33	**6.6**	2.94	**77**	164	**33**	6.29	**69**	108	**22**
BONSAI (90%)	25.84M	IA	874	–	513.47	–	2023	–	0.75	–	2023	–	7.15	–	2023	–	39.94	–	2023	–
		800	516	**59**	401.06	**78**	0.37	**0.02**	1.48	**196**	112	**5.5**	9.18	**128**	675	**33**	34.11	**85**	24	**1.2**
VISMALE (90%)	28.08M	IA	950	–	542.38	–	2200	–	0.85	–	2200	–	9.17	–	2200	–	45.08	–	2200	–
		800	561	**59**	434.52	**80**	0.4	**0.02**	1.72	**202**	122	**5.5**	11.31	**123**	684	**31**	37.24	**83**	14	**0.6**
FOOT (90%)	31.24M	IA	1054	–	600.89	–	2452	–	1.16	–	2452	–	11.10	–	2452	–	42.72	–	2452	–
		800	619	**59**	531.01	**88**	0.4	**0.01**	1.91	**165**	120	**5**	13.72	**124**	725	**30**	39.97	**94**	15	**0.6**
MAUI	4.00M	IA	160	–	23.97	–	199	–	0.07	–	199	–	4.70	–	199	–				
		300	99	**62**	17.79	**74**	0.04	**0.02**	0.27	**395**	54	**27**	3.79	**81**	31	**15**				
BAIA	8.32M	IA	333	–	48.66	–	411	–	0.08	–	411	–	7.90	–	411	–				
		200	209	**63**	35.53	**73**	0.03	**0.01**	0.48	**598**	119	**29**	5.98	**76**	118	**29**				
PUGET	19.46M	IA	779	–	114.31	–	952	–	0.59	–	952	–	17.59	–	952	–				
		600	484	**62**	86.94	**76**	0.08	**0.01**	1.52	**259**	248	**26**	13.97	**79**	198	**21**				

14(6):1619–1626, 2008.

[14] A. Gyulassy, N. Kotava, M. Kim, C. Hansen, H. Hagen, and V. Pascucci. Direct feature visualization using Morse-Smale complexes. *IEEE Transactions on Visualization and Computer Graphics*, PP(99):–, 2011.

[15] H. C. King, K. Knudson, and M. Neza. Generating discrete Morse functions from point data. *Experimental Mathematics*, 14(4):435–444, 2005.

[16] T. Lewiner, H. Lopes, and G. Tavares. Applications of Forman's discrete Morse theory to topology visualization and mesh compression. *IEEE Transactions on Visualization and Computer Graphics*, 10(5):499–508, Sept. 2004.

[17] J. Milnor. *Morse Theory*. Princeton University Press, New Jersey, 1963.

[18] I. Navazo. Extended octree representation of general solids with plane faces: Model structure and algorithms. *Computer & Graphics*, 13(1):5–16, 1989.

[19] R. Nelson and H. Samet. A population analysis for hierarchical data structures. In *Proc. ACM SIGMOD Conference*, pages 270–277, 1987.

[20] V. Robins, P. J. Wood, and A. P. Sheppard. Theory and algorithms for constructing discrete Morse complexes from grayscale digital images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(8):1646–1658, Aug. 2011.

[21] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. The Morgan Kaufmann series in computer graphics and geometric modeling. Morgan Kaufmann, 2006.

[22] H. Samet and R. Webber. Storing a collection of polygons using quadtrees. *ACM Transactions on Graphics*, 4(3):182–222, 1985.

[23] K. Weiss and L. De Floriani. Simplex and diamond hierarchies: Models and applications. *Computer Graphics Forum*, 30(8):2127–2155, 2011.

[24] K. Weiss, R. Fellegara, L. De Floriani, and M. Velloso. The PR-star Octree: A spatio-topological data structure for tetrahedral meshes. In *Proceedings ACM SIGSPATIAL GIS*, GIS '11, pages 92–101. ACM, November 2011.