Efficient Computation and Simplification of Discrete Morse Decompositions on Triangulated Terrains

Riccardo Fellegara
DIBRIS
University of Genova
Genova, Italy
riccardo.fellegara@unige.it

Federico Iuricich
Department of Computer Science and UMIACS
University of Maryland
College Park (MD), USA
federico.iuricich@gmail.com

Leila De Floriani
DIBRIS
University of Genova
Genova, Italy
leila.defloriani@unige.it

Kenneth Weiss
Lawrence Livermore National Laboratory
Livermore (CA), USA
kweiss@llnl.gov

ABSTRACT

We consider the problem of efficient computing and simplifying Morse complexes on a Triangulated Irregular Network (TIN) based on discrete Morse theory. We develop a compact encoding for the discrete Morse gradient field, defined by the terrain elevation, by attaching it to the triangles of the TIN. This encoding is suitable to be combined with any TIN data structure storing just its vertices and triangles. We show how to compute such gradient field from the elevation values given at the TIN vertices, and how to simplify it effectively in order to reduce the number of critical elements. We demonstrate the effectiveness and scalability of our approach over large terrains by developing algorithms for extracting the cells of the Morse complexes as well as the graph joining the critical elements from the discrete gradient field. We compare implementations of our approach on a widely-used and compact adjacency-based topological data structure for a TIN and on a compact spatio-topological data structure that we have recently developed, the PR-star quadtree.

Categories and Subject Descriptors

E.1 [**Data**]: Data Structures—*Graphs and networks*; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling; I.3.6 [**Computer Graphics**]: Methodology and Techniques—*Graphics data structures and data types*

Keywords

Discrete Morse theory, morphological simplification, spatio topological data structure

1. INTRODUCTION

Morphological methods for terrain analysis are rooted in Morse theory [23], which is the basis for defining decompositions of the domain of a scalar field into regions of influence of its critical points, called *Morse decompositions*. However, Morse theory applies to smooth functions, while, in practical applications, we deal with terrains 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 [16] extends Morse theory to simplicial complexes (triangle meshes in the 2D case). In addition to its theoretical contributions, this discrete formulation has practical applications in the development of robust discrete algorithms, which overcome the intrinsic limitations of previous algorithms for Morse decompositions.

Our work concentrates on computing discrete Morse decompositions on *Triangulated Irregular Networks (TINs)* discretizing terrains. To this aim, we compute a discrete Forman gradient compatible with the elevation values of the vertices of the TIN. The discrete gradient field is defined not only at the vertices, but also at the edges and triangles of the triangle mesh. However, a data structure for the mesh encoding all such entities would be too verbose. To overcome this problem, we introduce a compact encoding for the Forman gradient which is only attached to the triangles, and is therefore suitable to be combined with any compact data structure for triangle meshes encoding only vertices and triangles, like the *Indexed data structure with Adjacencies (IA)* [25]. Moreover, we propose an implementation for the PR-star quadtree [29], which derives the local connectivity of the mesh through a spatial index on the mesh geometry, and we show the advantages of the PR-star quad-tree in terms of storage and of efficiency in computing the Forman gradient and the associated discrete Morse decompositions based on a TIN.

A relevant issue when describing a terrain of large-size through a morphological representation, like a Morse decomposition, is the large size of the resulting terrain segmentation, due to the presence of noise and to uninteresting terrain features. A central contribution of our work is an algorithm for simplifying a morphological representation of large-size terrains based on the *cancellation* operator, which removes two critical points connected by an integral-line [22]. Moreover, simplification allows the extraction of morphological features, such as the cells of the Morse decomposition, the critical net connecting the critical points of the terrain, at different levels of topological accuracy.

The remainder of this paper is organized as follows. In Section 2, we discuss some background notions on smooth and discrete Morse theories. In Section 3, we discuss related work on computing Morse decompositions and on spatial indexes. In Section 4, we describe the compact Forman gradient encoding and its implementation on the IA and PR-star data structures, while, in Section 5, we provide a constructive definition of Morse complexes in the discrete case. In Section 6, we describe the simplification algorithm, and its implementation on the IA and PR-star data structures, while, in Section 7, we present an experimental evaluation of the results. Finally, in Section 8, we draw some concluding remarks and discuss current and future developments.

2. BACKGROUND NOTIONS

Morse theory [22,23] studies the relationships between the topology of a shape and the critical points of a scalar function defined on it. We consider a C^2-differentiable real-valued function f defined over a domain $M \subseteq \mathbb{R}^2$. A point $p \in \mathbb{R}^2$ is a *critical point* of f if and only if the gradient $\bigtriangledown f$ of f vanishes on p, i.e., $\bigtriangledown f(p) = 0$. We consider the Hessian matrix of f, denoted as $Hess(f)$, which is the matrix of the second-order partial derivatives of function f. If the determinant of $Hess_p(f)$ in p is not null, then p is a *non-degenerate critical point*. The number of negative eigenvalues of $Hess_p(f)$ is called the *index* of critical point p. In the 2D case, the critical points are minima, maxima and saddles, and they are critical points of index 0, 2 and 1, respectively.

We define a scalar field as a pair $\mathbb{M} = (M, f)$ where M is a domain in $\mathbb{R}^2$ and f is a C^2-differentiable real-valued function defined on M (the elevation in case of terrains), which associates a real value with each point of M. Function f is said to be a *Morse function* if and only if all its critical points are not degenerate. An *integral line* of a function f is a maximal path everywhere tangent to the gradient of f. Integral lines that converge to a critical point p of index i form the so-called called the *descending i-manifold* of p. Integral lines that originate from a critical point p of index i form the so-called *ascending* $(2-i)$*-manifold* of p. The ascending/descending manifolds are pairwise disjoint and decompose the domain of M into open cells. The collection of all the descending(ascending) manifolds forms the *descending(ascending) Morse complex.*

Discrete Morse theory [16] is a combinatorial counterpart of Morse theory: it defines a so-called discrete Morse function on all the cells of a cell complex. For the sake of simplicity, we will briefly review this theory for triangle meshes. A triangle mesh Σ is composed of triangles, edges and vertices, that we call 2-, 1- and 0-simplexes, respectively. Recall that a *k-dimensional simplex* or, *k-simplex* σ, spanned by a geometrically independent set $S = \{v_0, v_1, ..., v_k\}$ in $\mathbb{R}^n$. is defined as the convex hull of the points of S.

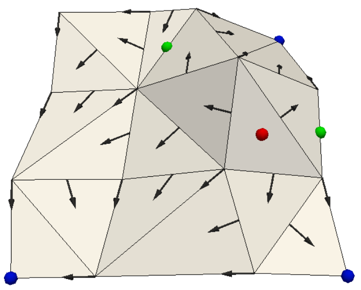

Figure 1: Example of a Forman gradient on a triangulated terrain. Red dots indicate critical triangles (maxima), green dots indicate critical edges (saddles) and blue dots critical vertices (minima).

A function $F : \Sigma \rightarrow \mathbb{R}$, defined on Σ, is a *discrete Morse function* (also called a *Forman function*) if and only if for every i-simplex $\sigma \in \Sigma$, all the $(i-1)$-simplexes on the boundary of σ have a lower function value than σ, and all the $(i+1)$-simplexes bounded by σ have a higher function value than σ, with at most one exception. If there is such an exception, it defines a pairing of cells, called a *gradient pair.* Otherwise, σ is a *critical* simplex of index p. A non-critical simplex σ can be paired either with a non-critical simplex bounded by σ or with one of its faces. The pair can be viewed as an *arrow* formed by a head (i-simplex) and a tail (($i-1$)-simplex). A simplex that is not a head or a tail of any arrow is a critical simplex.

A *V-path* is a sequence of simplex pairs $[\sigma_0, \tau_0, ..., \sigma_i, \tau_i, ..., \sigma_q, \tau_q]$ such that σ_i and σ_{i+1} are on the boundary of τ_i and (σ_i, τ_i) are paired simplexes, where $i = 0, ..., q$. The collection of all paired and critical simplexes of Σ forms a *discrete Morse gradient* (also called a *Forman gradient*) V if there are no closed V-paths in V, i.e., if all the V-paths are acyclic. In Figure 1, we show an example of discrete Morse gradient on a simplified terrain dataset. In the combinatorial setup of discrete Morse theory, V-paths correspond to the integral lines of the discrete Morse function F defined on Σ. We will call *separatrix V_j-path* any V-path of the following form: $[\tau, \sigma_0, \tau_0, ..., \sigma_i, \tau_i, ..., \sigma_q, \tau_q\sigma]$, where τ and σ are two critical simplexes of dimension j and $(j-1)$, respectively. In a triangle mesh Σ, we have separatrix V_1-paths connecting a critical edge (corresponding to a saddle) to a critical vertex (corresponding to a minimum), and separatrix V_2-paths connecting a critical triangle (corresponding to a maximum) to a critical vertex (corresponding to a saddle).

3. RELATED WORK

There have been several approaches to extend the results of Morse theory and to represent Morse and Morse-Smale (MS) complexes in the discrete case. Besides discrete Morse theory [16], reviewed in Section 2, another approach, discussed in [12,13], is based on Banchoff's extension of Morse theory to piecewise-linear manifolds and functions [1]. Surveys of algorithms for computing Morse and Morse-Smale complexes in 2D and 3D either based on the piece-wise linear Morse theory or on a watershed approach can be found in [2,7].

Recently, a lot of attention has been devoted to algorithms rooted in discrete Morse theory [16], which has led to the formulation of robust discrete algorithms [18,26,28] for computing Morse and MS complexes, overcoming the intrinsic limitations of previous techniques. Algorithms based on discrete Morse theory have been developed for regular grids [18,26] or for tetrahedral meshes [30]. A parallel algorithm for computing 2D Morse-Smale complexes for large 2D structured meshes is presented in [28]. Algorithms that produce more accurate geometry and, thus, the correct connectivity for the MS complex are discussed in [19].

A variety of hierarchical spatial indexes have been proposed for points, polygonal maps, 3D objects and triangle meshes [27]. In [9], a family of spatial indexes for tetrahedral meshes, called *tetrahedral trees*, has been introduced which generalize similar data structures for maps and triangle meshes to 3D. Such indexes contain the geometrical entities only in their leaf nodes and, thus, the shape of the tree is independent of the order in which the entities are inserted.

A fundamentally and conceptually different data structure for tetrahedral meshes, called the *PR-star octree* [29], uses the spatial embedding of the mesh to index its topological connectivity (see Section 4.1). This data structure has been used to extract morphological features from terrain and volume datasets in [8, 30] using a streaming implementation of Robins et al.'s algorithm for Forman gradient computation [26]. Other approaches propose localized computations by using a spatial index to reduce memory requirements for out-of-core [5], or memory intensive mesh processing [10], or by developing a reduced data structure for a simplicial mesh [3]. Cignoni et al. [5] introduce an external memory spatial data structure to support compact out-of-core processing of large triangle meshes.

4. ENCODING A TIN AND A FORMAN GRADIENT

When a terrain is represented as a *Regular Square Grid* (RSG), a Forman gradient V can be implicitly encoded on such representation as a bit vector based on the same indexing as the one used for the cells of the grid [17]. When using simplicial meshes, the large majority of the algorithms for computing the discrete Morse complexes, also in the context of homology and persistent homology computation, encode the simplicial mesh as an *Incidence Graph* (IG) [11]. A more compact version of the IG, specific for simplicial meshes, is the *Simplified Incidence Graph (SIG)* [15] which, in the case of a triangle mesh Σ, encodes all the simplexes of Σ and the *boundary relations* between a triangle and its bounding edges and an edge and its extreme vertices plus, for each edge, the indexes of its two incident triangles and, for each vertex, one edge incident in such vertex.

Data structures for simplicial meshes, which encode only the vertices and the maximal simplexes, like the *Indexed data structure with Adjacencies* (IA) [24,25], have been shown to be much more compact [4,14]. For a triangle mesh Σ, the IA data structure encodes only the vertices and the triangles, and, for each triangle σ, the indexes of its three vertices and of its three adjacent triangles and, for each vertex v, its coordinates plus the index of one triangle incident in v. As shown in [14], the IA data structure occupies 1.8 more space on average than an IG for a triangle mesh, and 1.4

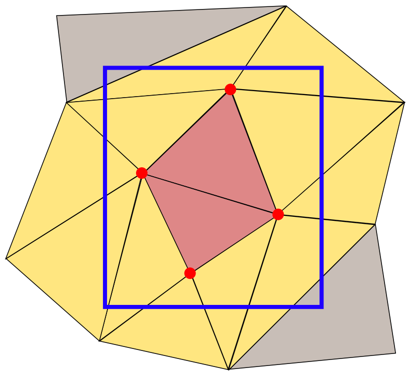

Figure 2: A leaf node in a PR-star quadtree (with bucket threshold $k_v = 4$) encodes a set of vertices and all triangles incident in such vertices. In brown the triangles that are completely indexed by the leaf (blue rectangle). In yellow, those that are incident into a vertex contained inside the leaf, and in grey those that geometrically intersect the leaf but are not incident into its internal vertices (these latter are not indexed by the leaf).

more space than a SIG. A more compact alternative to the IA data structure is provided by the PR-star quadtree, which still encodes only vertices and triangles, but uses an entirely different approach to handle mesh connectivity and topological relations.

4.1 The PR-star quadtree

The *PR-star quadtree*, a 2D version of the PR-star octree introduced in [29], uses the spatial index induced by a quadtree to efficiently generate local application-dependent topological data structures at runtime. The PR-star quadtree is based on the *Point Region quadtree (PR quadtree)*, a spatial index for point data [27]. The domain decomposition defined by a PR-quadtree is controlled by a single parameter, that we denote as 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 PR-quadtree causes the leaf to split and its indexed points to be redistributed among its four children. Thus, the domain decomposition induced by a PR-quadtree is independent of the insertion order of its points.

The *PR-star quadtree* for a triangle mesh Σ encodes the vertices and the triangles of Σ and consists of: (a) an array P of vertices, encoding the geometry of Σ; (b) an array T of indexed triangles, where each element is encoded in terms of the indices of its three vertices within P; and (c) an augmented PR quadtree N, whose leaf nodes index a subset of vertices from P, and all the triangles in T incident in these vertices. An example of PR-star quadtree is shown in Figure 2.

Here, use a more compact representation for the leaves of the PR-star quadtree compared with the one in [29] by exploiting the spatial locality provided by the quadtree through a reindexing of arrays P and T. Besides the hierarchical information associated with the quadtree (i.e., the pointers from a node to its parent and to its children), each leaf node

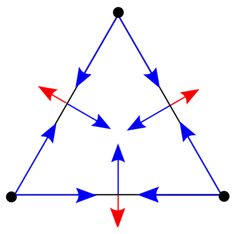

Figure 3: Set of arrows inside per triangle σ. Blue arrows indicate pairings between simplexes belonging to the boundary of σ (and possibly σ itself). Red arrows indicate pairings between the edges of σ and its adjacent triangles.

n_L encodes: the range of indices v_{start} and v_{end} in P of the vertices contained in n_L; the range of indices t_{start} and t_{end} in T of the triangles that are completely contained in n_L; and a pointer to the list of the remaining triangles from T incident in these vertices, i.e. each such triangle has at least one vertex inside and one outside the domain of n_L.

The IA and the PR-star data structures share the same encoding for the triangle-vertex connectivity of the mesh, i.e., the boundary relation between each triangle and its vertices, which requires $3|T|+3|P| \approx 9|P|$ items, since $|T| \approx 2|P|$, as a consequence of Euler formula. The IA data structure requires additional $3|T| + |P| \approx 7|P|$ items for the other topological relation encoded, that we call its *topological overhead*. Conversely, each node of a PR-star quadtree requires $9|N|$ items for the spatial indexing, and $\chi|T|$ for storing the triangle lists, where χ denotes the average number of quadtree nodes in which a triangle appears (i.e., $1 \leq \chi \leq 3$). This gives a total topological overhead for the PR-star data structure of $\chi|T| + 9|N|$. Based on our experiments, we approximate $\chi \approx 1.5$, and the number of quadtree nodes as $|N| \approx |P|/k_v$. Thus, the topological overhead for the PR-star is equal to $\chi|T| + 9|N| \approx 1.5|T| + (9/k_v)|P| \approx 4|P|$.

4.2 An Encoding for the Forman Gradient

We describe here an encoding for the Forman gradient V associated with a triangle mesh Σ, in which information regarding the Forman gradient V are attached only to the triangles. The encoding associates with each triangle σ in Σ a subset of the vector pairs involving its faces, and thus, it is stored globally for both the IA and PR-star data structures. Specifically, σ encodes all the vectors pairs, namely (τ_1, σ'), corresponding to an arrow from an edge τ_1 to a triangle σ' (red arrows in Figure 3) and (τ_0, τ_1), corresponding to an arrow from a vertex τ_0 to an edge τ_1 (blue arrows in Figure 3).

In a triangle mesh, a triangle has $\binom{3}{i+1}$ faces of dimension i, and each face has $(i+1)$ simplexes of dimension $(i-1)$ on its boundary. Thus, there are

$$\sum_{i=1}^{2} \binom{3}{i+1} \cdot (i+1) = 3 \cdot 2 + 1 \cdot 3 = 9$$

possible gradient pairs in the restriction of V to σ. Adding the three additional gradient pairs from an edge of σ to an adjacent triangle gives a total of 12 possible gradient pairings. Since each such pairing within a local frame encodes a single bit of information (i.e. the presence or absence of that particular pairing), each local frame can be encoded using 12 bit flags per triangle. This bit flag representation simplifies testing for the presence of vector pairings as well as updates to the discrete vector field. The restrictions imposed by discrete vector fields imply that there are significantly fewer valid local frame configurations than the possibilities provided by the bit flag representation. In 2D, we have 12 arrows for a total of $2^{12} = 4,096$ cases. However, since we are considering a Forman gradient, we have only 97 valid cases for a triangle. Thus, we can encode all the possible configurations by using only one byte per triangle. This has been generalized to 3D [30] and higher dimensions [21].

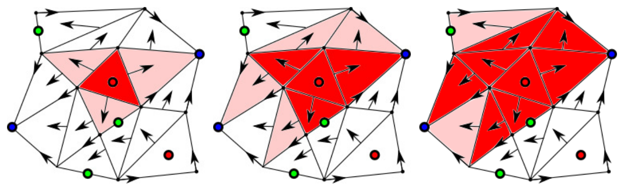

Figure 4: Example of the descending 2-manifold computed by starting from the critical triangle (in red).

We compute the discrete Morse gradient starting from the original mesh Σ and the elevation values given at its vertices. We have developed an implementation of the algorithm by Robins et al. [26], originally defined for cubical cell complexes and applied for persistent homology computation of 2D and 3D images, both for the IA and the PR-star data structures [8]. The algorithm outputs a list C of the critical simplices as well as the discrete gradient field encoded as a collection of *arrows* from a vertex to its paired edge, and from an edge to its paired triangle. The gradient is generated via homotopic expansions of the *lower star* of each vertex of the input mesh. Recall that the *lower star* of a vertex v consists of the simplices σ (edges and triangles) incident in v such that $\max_{p \in \sigma} f(p) = f(v)$. Since pairings occur only between simplices in the same lower star, each lower star can be treated independently.

5. EXTRACTING THE DISCRETE MORSE COMPLEXES

In this section, we provide a constructive definition of the cells of the discrete Morse complexes (i.e. the descending and ascending manifolds in the discrete case), and the critical net in terms of the triangles and vertices of the TIN. In Section 7, we briefly discuss how to compute such morphological features on the IA and the PR-star data structures.

A *descending 2-manifold* (see Figure 4) for a maximum p_{max} is a collection of *triangles* of Σ obtained by considering the critical triangle τ corresponding to p_{max} and following the (edge,triangle) arrows in V starting from the boundary edge σ of τ such that (σ, τ) is in V. Dually, an *ascending 2-manifold* (see Figure 5) for a minimum p_{min} is a collection of *vertices* of Σ obtained by starting from the critical vertex v and following the (vertex,edge) arrows in V.

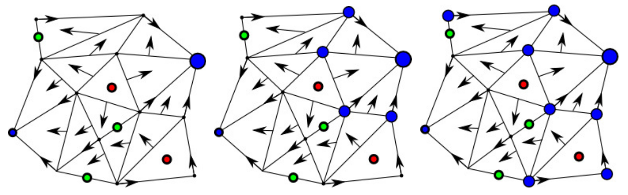

Figure 5: Example of the ascending 2-manifold computed from the critical vertex (in light blue).

A *descending 1-manifold* for a saddle p_{sad} is a collection of *edges* of Σ obtained by considering the critical edge e corresponding to p_{sad} and following the (vertex,edge) arrows in V starting from the vertex v on the boundary of e such that pair (v, e) is in V. Similarly, an *ascending 1-manifold* is a collection of *edges* of Σ and obtained by starting from an edge e corresponding to a saddle $p_{sad}$and following (edge,triangle) pairs of V. Note that each edge of Σ can be indexed either through its two extreme vertices or through its two incident triangles.

For extracting the descending 2-manifolds and the ascending 1-manifolds we need, for each edge e, the two triangles sharing e (*Edge-Triangle (ET) relation*), while for extracting the ascending 2-manifolds and the descending 1-manifold we need, for each vertex v, if exists, a triangle that contains the edge paired with v (*partial Vertex-Triangle (VT*) relation*). In the IA data structure, the partial VT relation is stored and the ET relation can be efficiently derived by the *Triangle-Triangle* adjacency relation stored. In the PR-star quadtree, the two topological relations are extracted when expanding each leaf node. The strategy in following the V-paths is different in the two data structures, since, in the PR-star quadtree, following the V-path can lead to a leaf that has not been visited yet (we refer to such paths as *dangling paths*). In this case, we stop the V-path following process and the information needed to complete the path are saved in an auxiliary data structure.

The union of the ascending and descending 1-manifolds forms *critical net*. The combinatorial structure of the critical net, the *Morse Incidence Graph (MIG)* [21], is a graph in which the nodes correspond to the critical points (simplices in Σ) and the arcs to the adjacencies of these points on the critical net. We observe that this is also the incidence graph representation of either Morse complexes, and thus it is computed by extracting all the manifolds in either one of the two Morse complexes.

6. SIMPLIFYING A FORMAN GRADIENT

A fundamental issue in terrain analysis is the complexity of available datasets, due to noise and the presence of uninteresting terrain features. In Morse theory an operator, called *cancellation*, has been defined, which removes critical points in pairs, thus simplifying the scalar field [22]. If $\mathbb{M} = (M, f)$ is a terrain, where f is a Morse function, a *cancellation* applied to two critical points p and q transforms f into a new Morse function g by removing p and q, and modifying the gradient field of f around the integral lines of p and q. In terms of Morse complexes, a cancellation produces the removal of two Morse cells, both in the ascending and descending Morse complexes, as well as a local modification of the incidence relations between the remaining Morse cells. If p and q are critical points of index $i + 1$ and i, respectively, *i-cancellation*(p, q) can be applied if and only if there is a unique integral line connecting p and q. As an effect of *i-cancellation*(p, q), p and q are removed and the integral lines originated/converging into them are modified. The set of integral lines converging at p or q before the cancellation are transformed into a set of integral lines converging to critical points of index $j > i$, that were the destination of integral lines starting at p before the cancellation; the set of integral lines that originated at q or p before the cancellation are transformed into a set of integral lines originating at critical points of index $k < i+1$ that were the origin of integral lines ending at q before the cancellation.

Working with a Forman gradient, there exists a discrete counterpart of the i-cancellation removing two critical simplexes in pairs [16]. Let us consider a Forman gradient V on mesh Σ and let p and q be two critical simplexes in V such that there exists exactly one gradient path from p to q. Then, there is another Forman gradient V' on Σ with the same critical simplexes with the exception of q and p. Moreover, V' coincides with V except along the unique gradient path from the boundary of p to q. V' is obtained from V by removing the critical $(i - 1)$-simplex q and i-simplex p and by reversing the arrows of the separatrix V_i-path $p, (\sigma_0, \tau_1), (\sigma_1, \tau_2), ..., (\sigma_i, \tau_i + 1), ..., (\sigma_n - 1, \tau_n), q$ connecting them. This produces the new V-path: (q, τ_n),-$(\sigma_n, \tau_{n-1}), ..., (\sigma_i, \tau_{i-1}), ..., (\sigma_0, p)$.

In Figure 6(a), we show an example of 1-cancellation between a maximum p and a saddle q. Cancellation works on the separatrix V-path, formed by triangles and edges, between the two critical simplexes (red arrows). After the 1-cancellation, p and q are removed from the Forman gradient and the arrows forming such V-path are reversed. Notice that, after the 1-cancellation, new V-paths are formed from the composition of the reversed path (red arrows) and the V-paths starting from the deleted maximum p (green arrows). Similarly, a 0-cancellation between a saddle and a minimum, illustrated in Figure 6(b), removes a critical edge q and a critical vertex p from the Forman gradient, which is performed by reversing the arrows on the V-path, formed by edges and vertices, between them (red arrows).

6.1 Simplification algorithm

We have defined a simplification algorithm, based on $i-$ *cancellation*, for simplifying the topology of a TIN, which is completely independent of the underlying data structure. A *persistence* value is associated with an $i - cancellation$ by considering the maximum elevation values of the two critical simplexes p and q deleted by the operator. The persistence of a pair of critical simplexes measures the importance of the pair. The objective of the simplification algorithm is to reduce the complexity of a scalar function by removing critical points which are due to the presence of noise or which are not relevant for the need of a specific application. Simplification is also applied when the size of the original Morse complex is too large for the available computation resources.

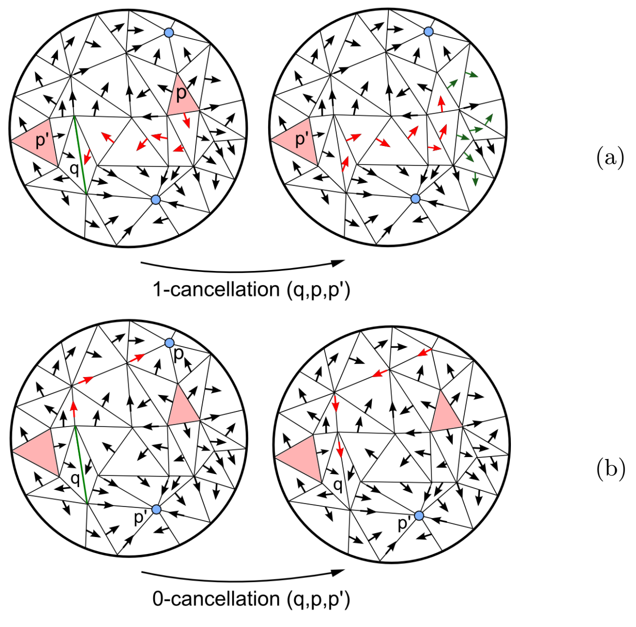

Figure 6: 1-cancellation (above) and 0-cancellation (below) removing two critical simplexes p and q from a Forman gradient.

Algorithm 1 SimplifyForman(θ,V,G)

```
Require: θ is a float value indicating the persistence threshold;
Require: G = (N, A) is the MIG describing the critical simplexes and their connections;
Require: V is the original Forman gradient;
 1: Q = ∅ // an empty priority queue
 2: // All the arcs are pushed into the priority queue based on their persistence value;
 3: for all arcs a ∈ A do
 4:     if a.persistence() < θ then
 5:         Q.push(a);
 6: while Q.isNotEmpty() do
 7:     s ← Q.pop() // The next simplification s is popped from the queue;
 8:     if isFeasibleSimpl(s) then
 9:         V.update(s); // swaps the arrows in the gradient
10:         a_new = G.update(s);
            // a_new are the new arcs created by the simplification
11:         for all arcs a ∈ a_new do
12:             if a.persistence() < θ then
13:                 Q.push(a);
14:     else
15:         break;
```

Algorithm 6.1 provides the pseudo-code description of the simplification algorithm. The stopping condition is determined by a persistence threshold θ, i.e., all the simplifications performed have a persistence value lower than θ. Function *SimplifyForman* requires three input parameters: the threshold θ, a Forman gradient V and the MIG G computed on Σ and on V.

In the initialization step, the arcs having a persistence value below threshold θ are inserted in a priority queue Q (rows 3-4-5), sorted according to their persistence. Each simplification s in Q, which removes two critical simplexes p and q, is considered feasible if there exists a single V-path between p and q, i.e., the two corresponding nodes are in G and they are connected by a single arc (function *isFeasibleSimpl*). If the simplification is feasible then:

- we update the gradient vector, by removing p and q and by swapping the arrows in the V-path between them (row 9);
- we update the MIG G by removing the two corresponding nodes from G and by creating for each saddle connected with p, different from q, a new arc connecting it with the other minimum/maximum connected with q and different from p (row 10).

Those new arcs are also inserted into Q if their persistence value is below θ.

6.2 Forman gradient simplification on the IA data structure

The persistence-based simplification algorithm *SimplifyForman* has been implemented in the IA data structure. The parts of the algorithms, which depend on the data structure, are the computation of the input MIG from mesh Σ and Forman gradient V, and the update of the Forman gradient when performing an i-cancellation. The MIG is computed by extracting the descending and ascending 1-manifolds. In the IA implementation we create a node for each critical simplex in V. Then, for each critical edge (saddle) s, we expand the descending and ascending 1-manifolds. For expanding the ascending 1-manifolds we follow, from each saddle s, the separatrix V_2-paths starting from the triangles in bounded by s (Edge-Triangle topological relation). For each critical triangle t reached by following the separatrix V_2-paths, a new arc in G is created connecting s to t. Similarly, to expand the descending 1-manifolds, we follow the separatrix V_1-paths starting from the boundary vertices of s (Vertex-Edge topological relation). For each critical vertex p reached by following the separatrix V_1-paths, a new arc in G is created connecting s to p. The update of the Forman gradient at each simplification step is performed by using the same topological relations.

6.3 Forman gradient simplification on the PR-star quadtree

Here, we describe the implementation of the simplification algorithm specific for the PR-star quadtree. The strategy described is entirely different since it exploits the modular structure of the PR-star quadtree. The basic paradigm for performing operations on a mesh encoded as a PR-star quadtree is to locally process the mesh in a streaming manner by iterating through the leaf nodes. For each leaf node, a local application-dependent data structure, which we refer to as an *expanded leaf node*, is generated and used to process the local geometry. After processing a leaf node, we discard this local structure and move to the next leaf node. For efficiency, we use an auxiliary cache based on the least-recent-used replacement policy that maintains a subset of expanded leaf nodes.

The whole simplification process is executed in those leaf nodes that contain at least one saddle. The following simplification steps are performed:

1. compute an *expanded leaf node* representation by extracting all the topological relations required;
2. compute a local MIG;
3. simplify the global gradient V and the MIG G by using the $SimplifyForman$ algorithm;
4. save the expanded leaf node representation in cache.

Recall that, when indexing a mesh Σ in a PR-star quadtree, we encode only the relation between each triangle of Σ and its bounding vertices. At run-time we compute an *expanded leaf node* data structure for the portion of Σ in a leaf node n_L, that we denote Σ_L. The data structure encodes also the edges of the mesh, the two triangles sharing a given edge (Edge-Triangle relation), and, for each indexed vertex v, a triangle that contains the edge paired with vertex v (partial Vertex-Triangle relation). To reduce the overall computational complexity, these relations are also cached.

Thanks to the intrinsic modularity of the PR-star tree, we have defined two different execution strategies (see step 2), which compute the MIG as a completely *local* data structure, or as a *global* data structure. In the *local* strategy, we construct a *coherent* local MIG for each leaf nod n_L: for each saddle inside n_L, we traverse the global gradient V to find all the minima and maxima connected to it. Each saddle is visited exactly once, since it is indexed in a single leaf node, i.e., the one that indexes the maximum vertex of the saddle. On the other hand, even if the *local* MIG is coherent, with respect to the indexed saddles, the minima and maxima outside node n_L are not considered„ since we do not visit all the V-paths going out of n_L. In the *global* approach, there is an execution pattern similar to the one implemented in the IA data structure, where we first compute a global MIG and then we execute the simplification procedure. Thus, in the *global* strategy, we skip step 2.

In step 3 we use persistence-based simplification algorithm *SimplifyForman*. The main difference with respect to the IA implementation is that the priority queue is local to the leaf node, i.e., the queue contains only those arcs that have an indexed saddle into the current leaf node. This leads to a different simplification order with respect to the IA implementation, which has a global queue.

During the local MIG construction and the gradient simplification, we could have a relatively intense navigation of the index, that can cause a series of cache lookup, leading to a cache-hit or a cache-miss. If we have a cache miss we have to expand the leaf node in order to continue the current operation, and then add the expanded leaf node representation to the cache.

7. EXPERIMENTAL RESULTS

We evaluate here the performance of the discrete gradient encoding, of the morphological feature extraction algorithms and of the gradient simplification algorithm. We present experiments on three triangle meshes, representing terrains, whose sizes vary from 4 to 19.5 million triangles. The hardware configuration used is an Intel i7 3930K CPU at 3.20Ghz with 64 GB of RAM.

7.1 Storage costs and gradient computation

We compare here the storage costs of the IA and PR-star data structures. We call the *base mesh* the information on the mesh encoded by both data structures, consisting of the vertex coordinates and the triangle-vertex connectivity, and *topological overhead* the remaining storage cost for each of the two structures. Compared with the storage cost of the base mesh, the topological overhead of the IA data structure is about 20% less, while the topological overhead of the PR-star tree is between 85% and 90% less. In terms of overall storage cost, which includes the cost of the base mesh, the maximal topological overhead, and the gradient encoding, the PR-star requires from 20% to 35% less memory than the IA data structure.

Our implementations of the gradient construction algorithm on the IA and PR-star data structures share the same encoding, but the PR-star quadtree can efficiently reconstruct the local connectivity for the entire sub-mesh indexed by a leaf node and, thus, it computes the gradient field requiring from 10% to 15% less time than the IA data structure (see Table 1).

7.2 Extraction of Morse complexes

Both data structures require a small amount of additional memory to extract Morse features. The IA requires a global queue to perform the graph traversal of the gradient field, while the PR-star quadtree uses a cache of quadtree nodes with expanded connectivity information as well as a list of *dangling paths* for each visited leaf node, plus a small local queue for the local graph traversal. As shown in Table 1, the additional storage in the IA data structure implementation is negligible on the descending and ascending extractions, while it is between 2% and 9% higher, with respect to the mesh and gradient encoding, when extracting the MIG. Conversely, the additional storage required by the the PR-star quadtree implementation is slightly higher, since it is between 2% and 10% more.

Generally speaking, the IA data structure outperforms the PR-star quadtree in extracting specific i-manifolds. For the descending 2-manifolds, the IA is from 2.5 to 4 times faster, and for the ascending 2-manifolds, it is from 8 to 12 times faster then the PR-star. In extracting descending 1-manifolds the PR-star requires from 40% to 80% more time with respect to IA data structure. Conversely, the PR-star quadtree is faster in extracting the MIG, for which it requires from 10% to 20% less time than the IA data structure. The rationale for this is that, during the extraction of a specific i-manifold, only a specific topological relation is involved and this that can be extracted from the IA structure more efficiently. On the contrary, when different topological relations are involved, such as during the MIG extraction (or the Forman gradient computation), the PR-star tree approach is more efficient since it amortizes the time required in computing the auxiliary topological relations in the leaf nodes.

7.3 Topological simplification

Since topological simplification is an operation used to remove the *noise* from a scalar field, we have selected three

Table 1: Timings (in seconds) and storage (expressed in MBs) for the implementations based on the PR-star tree and the IA. *Mesh* shows the storage cost for the data structure encoding the mesh. *Connect.* shows the topological connectivity overhead of the structures, which is compared respect to the base mesh requirements. *Max total* shows the maximum storage requirements, with which we compare PR-star and IA.

Data	$\lvert T\rvert$	k_v	Storage					Timings											
			Mesh	Connect.		Max total		Gradient		Descend. 2		Descend. 1		Ascend. 2		Ascend. 1		MIG	
				tot	%	tot	%	tot	%	tot	%	tot	%	tot	%	tot	%	tot	%
		IA		57	**83**	129	–	26.0	–	4.74	–	1.62	–	1.86	–	9.42	–	11.0	–
MAUI	4.0M	200	69	11	**16**	87	**68**	21.5	**83**	17.2	**362**	2.50	**154**	23.5	**1264**	8.21	**87**	10.0	**91**
		400		7	**10**	79	**79**	22.2	**85**	17.9	**377**	2.90	**179**	19.1	**1029**	8.41	**89**	10.3	**93**
		IA		119	**83**	264	–	53.1	–	9.90	–	3.27	–	3.81	–	8.16	–	11.4	–
BAIA	8.3M	100	143	22	**16**	172	**65**	44.4	**84**	32.7	**331**	4.65	**142**	46.7	**1225**	7.15	**88**	9.07	**79**
		300		14	**9**	184	**70**	46.0	**87**	40.2	**406**	5.34	**163**	38.9	**1020**	8.39	**103**	10.7	**94**
		IA		278	**83**	642	–	128	–	24.0	–	8.82	–	9.70	–	31.5	–	40.3	–
PUGET	19.5M	400	334	35	**10**	413	**64**	111	**87**	69.6	**291**	13.5	**153**	85.3	**879**	24.7	**78**	31.0	**77**
		800		29	**9**	430	**67**	113	**88**	65.5	**273**	14.4	**163**	76.2	**785**	26.6	**84**	31.2	**77**

Table 2: Simplification timings (expressed in seconds), storage (expressed in MBs) and ratio for the implementations based on the PR-star tree and the IA. The *storage* is compared respect to the base mesh requirements shown in Table 1. $\lvert S\rvert$ colum shows the saddle number in the model. *Ratio* columns show the number of simpification executed. *Local* columns show the timings obtained using the *local* PR-star strategy, while *Global* columns show the timings obtained with a *global* PR-star strategy.

Data	$\lvert S\rvert$	k_v	Storage				Timings														
							Low noise removal					Medium noise removal					High noise removal				
			Local		Global		Ratio	Local		Global		Ratio	Local		Global		Ratio	Local		Global	
			tot	%	tot	%		tot	%	tot	%		tot	%	tot	%		tot	%	tot	%
		IA	61	**88**			6.29K	10.4				12.5K	10.6				18.1K	10.6			
MAUI	25.1K	200	15	**22**	18	**27**	5.12K	17.4	**167**	16.0	**154**	12.0K	20.1	**190**	16.0	**152**	18.1K	24.3	**230**	16.2	**153**
		400	23	**34**	26	**38**		17.7	**170**	15.5	**149**		20.3	**192**	15.7	**149**		23.9	**226**	16.2	**153**
		IA	121	**85**			4.34K	9.67				8.82K	9.73				12.8K	9.76			
BAIA	17.4K	100	27	**19**	29	**20**	2.54K	14.1	**146**	18.1	**187**	7.74K	14.6	**150**	18.1	**186**	12.3K	15.5	**158**	18.2	**186**
		300	30	**21**	32	**23**		16.9	**174**	21.3	**220**		17.4	**179**	21.5	**221**		18.0	**184**	21.8	**224**
		IA	309	**92**			56.8K	38.7				111K	39.9				170K	40.7			
PUGET	227K	400	49	**15**	79	**24**	45.5K	51.2	**133**	56.5	**146**	101K	56.1	**141**	60.7	**152**	163K	66.8	**164**	61.3	**150**
		800	66	**20**	96	**29**		56.2	**145**	59.6	**154**		61.3	**154**	64.1	**161**		70.2	**172**	66.0	**162**

levels of noise removal, choosing three different persistence thresholds (for a *lower*, *medium* and *larger* noise removal). Table 2 shows the results of these experiments. Since the algorithms developed for the IA data structure and for the PR-star quadtree use a different simplification order (a global versus a local priority queue), we obtain different simplification sequences for the same value of the persistence threshold. The order in the simplification influences the number of simplifications that can be executed during a simplification run. The two PR-star quadtree implementations (with local or global MIG) obtain similar simplification ratios, and both always execute fewer simplifications with respect to the IA implementation. This difference is more noticeable for lower threshold values.

The timings shown in Table 2 consider all the simplification steps, i.e., for the IA algorithms and the *global* PR-star strategy, we count the time required for the MIG computation and for simplification, while, for the *local* PR-star strategy, we consider the simplification procedure, which encompasses the local MIG computation. The IA implementation is always faster than both PR-star implementations, requiring from half the time to 40% less time for all thresholds. In the IA implementation, the simplification timings corresponds to the 1% of the time requirements, while the remaining time is required by the MIG computation. The same holds also for the *global* PR-star implementation, for which half of the time is spent in the MIG computation and almost the other half is required by the extraction of the topological relations and by cache management.

Comparing now the two PR-star implementations, we can see that the *local* strategy is faster on BAIA dataset, on which it requires from 15% to 20% less time, regardless of the threshold values, and on PUGET dataset for lower and medium thresholds, requiring from 5% to 10% less time. Conversely, the *global* strategy is faster on MAUI dataset, on which it requires from 10% to 50% less time, increasing the saving for the highest threshold; and on PUGET dataset, on which, for the highest threshold, requires from 5% to 10% less time. These results denote that the *local* strategy cannot amortize the local MIG extraction and the successive local simplifications when the tree navigation is intense, and when the persistence thresholds become larger.

The most compact approach is the *local* PR-star strategy, which requires a fraction of the storage required for the *MIG*, with respect to the IA and the *global* PR-star strategy, and a fraction of the storage for the priority queue, with respect to the IA data structure. Thus, with respect to the IA structure, it requires from 60% to 85% less memory, and with respect to the *global* PR-star strategy, it requires from 2% (on BAIA) to 10% (on PUGET) less space. Moreover, the *global* PR-star strategy, even if it explicitly stores a global *MIG* structure, it is still more compact than the IA data structure as it requires from 55% to 75% less memory.

8. CONCLUDING REMARKS

We have presented an efficient tool for computing and simplifying discrete Morse decompositions on triangulated terrains. We have described a compact encoding of a discrete Morse gradient field using the *local frame* representation, which associates information with the triangles. This encoding makes the IA data structure a very compact alternative to the other data structures, like IG or SIG, which are the common encoding of cell and simplicial complexes endowed with a Forman gradient. We have also shown that, at the expense of some additional computation, we can achieve greater savings by using the PR-star quadtree. We have developed a modular approach to the simplification of the Forman gradient based on the PR-star quadtree and we have compared it with a global simplification algorithm we have developed based on the IA data structure.

Both our gradient encoding and our formulation of topological features are independent of the data structure used to encode the mesh (as long as it encodes the vertices and the triangles), of the method in which the gradient field is computed and of the algorithms used for feature extraction. Thus, we anticipate our approach benefiting from theoretical advances in data structures on the one hand and in computational topology on the other. Specifically, we are developing a framework for homology computation on arbitrary-dimensional shapes discretized as simplicial meshes for machine learning applications. This approach is based on the discrete Morse decomposition and on the generalization of the IA data structure to such meshes. Our next step will be to develop a dimension-independent version of the PR-star tree and to perform the homology computation on it, where we expect an even larger saving in space and computation times.

We are currently implementing an algorithm for persistence-based simplification, which is based on the simplification operators in [6], expressed in terms of gradient field simplification. Persistence-based simplification will allow us to extract morphological features, such as the various manifolds, the extrema graphs (which are formed by the ascending and descending 1-manifolds) and the 1-skeleton of the MS complex at different levels of persistence, as in [20]. It can also remove the need for the preprocessing mesh simplification step (as discussed in Section 7).

In our future work, we plan to use a modified PR-star tree to encode the extracted i-manifolds, or the MS cells, which would allow us to reconstruct the topological connectivity of the various complexes as well as to efficiently perform spatial queries on them. Our approach can also be extended to time-varying datasets defined on simplicial meshes, and to tetrahedral shapes in 4D space (such as isosurfaces of time-varying fields), since the PR-star tree, the gradient encoding and the feature extraction are all dimension-independent.

Acknowledgments

This work has been partially supported by the US 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. Terrain datasets are courtesy of the Virtual Terrain Project (VTP) (BAIA and MAUI) and 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] D. Canino, L. De Floriani, and K. Weiss. IA*: An adjacency-based representation for non-manifold simplicial shapes in arbitrary dimensions. *Computers & Graphics*, 35(3):747–753, 2011.

[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 and L. De Floriani. Dimension-independent simplification and refinement of Morse complexes. *Graphical Models*, 73(5):261–285, 2011.

[7] L. Čomić, L. De Floriani, and F. Iuricich. Modeling three-dimensional morse and morse-smale complexes. In M. BreuÃ§, A. Bruckstein, and P. Maragos, editors, *Innovations for Shape Analysis*, Mathematics and Visualization, pages 3–34. Springer Berlin Heidelberg, 2013.

[8] L. De Floriani, R. Fellegara, F. Iuricich, and K. Weiss. A spatial approach to morphological feature extraction from irregularly sampled scalar fields. In *Proceedings ACM SIGSPATIAL International Workshop on GeoStreaming*, IWGS '12, pages 40–47, November 2012.

[9] 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.

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

[11] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer Verlag, Berlin, 1987.

[12] H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Morse-Smale complexes for piecewise linear 3-manifolds. In *Proceedings Symposium on Computational Geometry*, pages 361–370. ACM, 2003.

[13] H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical Morse complexes for piecewise linear 2-manifolds. In *Proceedings Symposium on Computational Geometry*, pages 70–79. ACM, 2001.

[14] L. D. Floriani and A. Hui. Data structures for simplicial complexes: An analysis and a comparison. In M. Desbrun and H. Pottmann, editors, *Symposium on Geometry Processing*, volume 255 of *ACM International Conference Proceeding Series*, pages 119–128. Eurographics Association, 2005.

[15] L. D. Floriani, A. Hui, D. Panozzo, and D. Canino. A dimension-independent data structure for simplicial complexes. In S. M. Shontz, editor, *IMR*, pages 403–420. Springer, 2010.

[16] R. Forman. Morse theory for cell complexes. *Advances in Mathematics*, 134:90–145, 1998.

[17] D. Günther, J. Reininghaus, H. Wagner, and I. Hotz. Efficient computation of 3D Morse-Smale complexes and persistent homology using discrete Morse theory. *The Visual Computer*, 28(10):959–969, 2012.

[18] A. Gyulassy, P. T. Bremer, B. Hamann, and V. Pascucci. A practical approach to Morse-Smale complex computation: Scalability and generality. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1619–1626, 2008.

[19] A. Gyulassy, P.-T. Bremer, and V. Pascucci. Computing Morse-Smale complexes with accurate geometry. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2014–2022, 2012.

[20] 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*, 18(9):1549–1562, 2012.

[21] F. Iuricich. Multi-resolution shape analysis based on discrete morse decompositions. In *PhD thesis, University of Genova – DIBRIS, Italy*, 2014.

[22] Y. Matsumoto. An introduction to Morse theory. Translations of mathematical monographs, volume 208. *American Mathematical Society*, 2002.

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

[24] G. M. Nielson. Tools for triangulations and tetrahedralizations and constructing functions defined over them. In G. Nielson, H. Hagen, and H. Müller, editors, *Scientific Visualization: Overviews, Methodologies and Techniques*, chapter 20, pages 429–525. IEEE Computer Society, 1997.

[25] A. Paoluzzi, F. Bernardini, C. Cattani, and V. Ferrucci. Dimension-independent modeling with simplicial complexes. *ACM Transactions on Graphics*, 12(1):56–102, January 1993.

[26] V. Robins, P. Wood, and A. Sheppard. Theory and algorithms for constructing discrete Morse complexes from grayscale digital images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1646–1658, August 2011.

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

[28] N. Shivashankar and V. Natarajan. Parallel computation of 3D Morse-Smale complexes. *Computer Graphics Forum*, 31(3):965–974, 2012.

[29] 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.

[30] K. Weiss, F. Iuricich, R. Fellegara, and L. De Floriani. A primal/dual representation for discrete morse complexes on tetrahedral meshes. In *Computer Graphics Forum*, volume 32, pages 361–370. Wiley Online Library, 2013.