Eurographics Italian Chapter Conference (2011)
A. F. Abate, M. Nappi, and G. Tortora (Editors)

A Decomposition-Based Approach to Modeling and Understanding Arbitrary Shapes

David Canino [1] and Leila De Floriani [1]

[1] Department of Computer Science, University of Genova, Genova, Italy

Abstract

Modeling and understanding complex non-manifold shapes is a key issue in shape analysis and retrieval. The topological structure of a non-manifold shape can be analyzed through its decomposition into a collection of components with a simpler topology. Here, we consider a representation for arbitrary shapes, that we call Manifold-Connected Decomposition *(MC-decomposition), which is based on a unique decomposition of the shape into nearly manifold parts. We present efficient and powerful two-level representations for non-manifold shapes based on the MC-decomposition and on an efficient and compact data structure for encoding the underlying components. We describe a dimension-independent algorithm to generate such decomposition. We also show that the MC-decomposition provides a suitable basis for geometric reasoning and for homology computation on non-manifold shapes. Finally, we present a comparison with existing representations for arbitrary shapes.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations

1. Introduction

Simplicial complexes are commonly used to represent objects in a variety of applications, including finite element analysis, solid modeling, animation, terrain modeling, and scientific visualization. They can model both manifold and non-manifold shapes, also with parts which are not uniform in their dimensions. Informally, a *manifold* (with boundary) is a compact and connected subset of the Euclidean space for which the neighborhood of each point is homeomorphic to an open ball, or to an open half-ball. Objects that do not fulfill this property at one or more points are called *non-manifold*, while they are called *non-regular*, if they also contain parts of different dimensions.

Existing modeling tools are generally designed to handle only shapes with a manifold domain, since they are topologically simpler [DFH05]. But non-manifold and non-regular objects arise in several applications. For example, in the idealization process for preparing an object for finite element simulations, regions representing beam or plate behavior are substituted with one-dimensional lines, or two-dimensional surfaces, respectively, resulting in objects that contain non-manifold singularities and parts of different dimensions. This process provides a representation of the input shape, which captures only its essential features. Non-manifold and non-regular shapes are usually discretized through simplicial, or cell complexes, and, thus, efficient representations for their discretized versions and topological methods for their analysis are required. The research on non-manifold and non-regular shapes in the literature has been focusing on effective data structures for modeling such shapes and on operators for manipulating it. On the other hand, it is important to decompose them into manifold (or almost manifold) parts, not only for the design of effective data structures, but also to be able to perform reasoning on non-manifold and non-regular shapes. This will allow us to deal with their intrinsic complexity.

Here, we represent arbitrary shapes discretized as simplicial complexes through a sound decomposition. A decomposition of an arbitrary shape should remove as many singularities as possible, and cuts it along singularities, without breaking it at manifold parts. We are considering a decomposition into so-called *manifold-connected* components, that we call *Manifold-Connected Decomposition (MC-decomposition)*. The basic concepts underlying this decomposition, but limited to 2D and 3D, have been introduced in [HDF07]. Here, we consider efficient two-level represen-

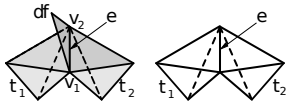
DOI: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2011/053-060

tations of the MC-decomposition for an arbitrary shape Σ, where the upper level consists of a collection of manifold-connected components, while the lower-level representation consists of a unique topological data structure describing each component in Σ. In this paper, we represent an arbitrary shape through the *Generalized Indexed data structure with Adjacencies (IA*)*, a dimension-independent and compact data structure for encoding arbitrary shapes [CDFW11]. We also provide dimension-independent algorithms for building the MC-decomposition of an arbitrary shape.

The remainder of the paper is organized as follows. Section 2 provides background notions on simplicial complexes and topological relations in a simplicial complex. Section 3 reviews related work on topological data structures, and on shape decomposition. In Section 4, we provide a brief description of the IA* data structure, while in Section 5, we describe data structures for encoding the MC-decomposition. In Section 7, we present some experimental results regarding the MC-decomposition. In Section 6, we briefly describe an application of the MC-decomposition regarding homological computation on arbitrary shapes. Finally, in Section 8, we draw some concluding remarks.

2. Background Notions

A Euclidean *simplex* σ of dimension k is the convex hull of a set V_σ of $k+1$ linearly independent points in the Euclidean space $\mathbb{E}^n$, with $0 < k \leq n$. We simply call a Euclidean simplex of dimension k as *k-simplex*. Here, k is the dimension of σ, and it is denoted as $dim(\sigma)$. Any Euclidean p-simplex σ', with $0 \leq p < k$, generated by a set $V_{\sigma'} \subseteq V_\sigma$ of cardinality $p+1$ is called a *p-face* of σ. Whenever no ambiguity arises, the dimensionality of σ' can be omitted, and σ' is called a *face* of σ. Any face σ' such that $\sigma' \neq \sigma$ is called a *proper face* of σ. A finite collection Σ of Euclidean simplices forms a Euclidean *simplicial complex* if and only if (i) for each simplex σ in Σ, all faces of σ are in Σ, and (ii) for each pair of simplices σ' and σ'', either $\sigma' \cap \sigma'' = \emptyset$, or $\sigma' \cap \sigma''$ is a common face. The maximum dimension d of simplices in Σ is called the *dimension* of the *simplicial d-complex* Σ. A subset of Σ that satisfies the definition of simplicial complex is a *sub-complex* of Σ. The *domain* of a Euclidean simplicial d-complex Σ embedded in $\mathbb{E}^n$, with $0 < d \leq n$, is the subset of $\mathbb{E}^n$ spanned by all simplices in Σ. The *combinatorial boundary* of a simplex σ is the set of all faces of σ. The *star* of a simplex σ, denoted with $St(\sigma)$, is the set of simplices λ in Σ that have σ as a proper face. In this case, simplices λ and σ are called *incident* simplices. Any simplex σ such that $St(\sigma) = \emptyset$ is called a *top* simplex. A simplicial d-complex Σ in which all top simplices are d-simplices is called *regular*. The *link* of a simplex σ, denoted as $Lk(\sigma)$, is the set of all faces of simplices in $St(\sigma)$, which are not incident at σ. Two k-simplices are *adjacent* if they share a k-face, with $k \leq 1$: in particular, two vertices are adjacent if they are both incident at a common edge. An h-path is a sequence of $(h+1)$-simplices $(\sigma_i)_{i=0}^k$ in a simplicial complex Σ such that two consecutive simplices in the sequence are h-adjacent. Two simplices σ' and σ'' are h-connected when there exists an h-path $(\sigma_i)_{i=0}^k$ such that σ' is a face of σ_0, and σ'' is a face of σ_k. A subset Σ' of a simplicial complex Σ is called *h-connected* if and only if any two simplices Σ are h-connected. Any maximal h-connected sub-complex of Σ is called an *h-connected component* of Σ. A regular $(d-1)$-connected simplicial d-complex Σ in which the star of all $(d-1)$-simplices consists of one or two simplices is called a *combinatorial pseudo-manifold*. A k-simplex in a simplicial d-complex Σ such that $Lk(\sigma)$ is homeomorphic to the $(d-k)$-sphere is *manifold*. A *manifold* simplicial complex is a pseudo-manifold where any simplex is manifold. Figure 1 shows two examples of non-manifold edges and vertices in a simplicial 3-complex: in both cases, $Lk(e)$ is formed by two connected components, and thus the edge e is not manifold. Also vertices v_1 and v_2 are non-manifold.



(a) (b)

Figure 1: *examples of non-manifold edges and vertices in a simplicial 3-complex. (a) A top triangle df, and two tetrahedra t_1 and t_2 are incident at the edge e. (b) None top triangle is incident at the edge e, shared by two tetrahedra t_1 and t_2.*

Topological relations are suitable for defining data structures describing simplicial complexes. They describe the connectivity among simplices, and the choice of which simplices and relations to be encoded determines the effectiveness of a data structure for specific applications. Let Σ be a simplicial d-complex, and σ be a p-simplex, with $0 \leq p \leq d$, then we define the following topological relations:

- for $0 \leq q \leq p-1$, *boundary relation* $\mathcal{R}_{p,q}(\sigma)$ consists of all q-faces of σ;
- for $p+1 \leq q \leq d$, *co-boundary relation* consists of all q-simplices incident at σ;
- for $p > 0$, *adjacency relation* consists of all p-simplices adjacent to σ.

We call *constant* any relation, which involves a constant number of entities. Relations, which involve a variable number of entities, are called *variable*. We consider an algorithm for retrieving a topological relation $\mathcal{R}$ to be *optimal* if it retrieves a given relation $\mathcal{R}$ in time linear with respect to the number of entities involved in $\mathcal{R}$.

3. Related Work

As discussed in [DFH05], there are several data structures for representing manifold shapes. Here, we restrict our attention to data structures for representing arbitrary shapes.

The first proposal for representing an arbitrary shape is the *Radial-Edge* data structure [Wei88], and its variants, as the *Partial Entity* [LL01], and *Loop Edge* [MMH01] data structures, that can be restricted to simplicial complexes. All such data structures are verbose and do not scale well to the manifolds [DFH05]. An alternative to previous data structures is the *Incidence Graph* (IG) [Ede87], a dimension-independent data structure for representing cell complexes. The *Simplified Incidence Graph* (SIG) [DFGH04], and the *Incidence Simplicial* (IS) [DFHPC10] are simplified versions of the IG data structure, restricted to simplicial complexes. They encode all simplices, and a subset of the co-boundary relations encoded in the IG data structure. However, such data structures may be too verbose, expecially in those applications that do not require all simplices. It is possible to minimize the subset of topological entities to be directly encoded. In this context, the *Indexed data structure with Adjacencies* (IA data structure) [PBCF93] is a pioneer work, but it is limited to pseudo-manifold complexes. The *Triangle-Segment* data structure [DFMPS04] is an adjacency-based representation for simplicial 2-complexes embedded in $\mathbb{E}^3$. It encodes all vertices and top simplices, specifically triangles and wire-edges. It is very compact and allows retrieving topological relations in optimal time. There are few representations for describing arbitrary shapes discretized as 3D simplicial complexes. Most such representations are limited to the manifold domain, like the *Facet-Edge* [DL89], and the *Handle-Face* [LT97] data structures. In [LLLV05], a scalable data structure for manifold tetrahedral complexes has been proposed, extending the *Corner Table* [RSS01]. Efficient extensions of the Corner Table have been recently proposed for encoding tetrahedral [GR09], and triangle meshes [GLLR11]. The *Non-manifold Indexed Data Structure with Adjacencies* (NMIA) [DFH03] is an adjacency-based representation for simplicial 3-complexes embedded in $\mathbb{E}^3$. It encodes all vertices and top simplices, specifically, tetrahedra, dangling faces and wire edges. It is compact, and supports the efficient retrieval of topological relations. In any case, our tests prove that the *Generalized Indexed with Adjacencies* (IA* data structure) [CDFW11] is the most cost effective than other dimension-independent representations, and is even slightly more compact than the existing dimension-specific ones.

In the literature, approaches to represent non-manifold shapes have been proposed based on the decomposition of the shape into manifold components. In [DS92], the authors proposed a decomposition of solid object into regular parts (r-sets), providing interesting topological information about an object. In [FR92], the authors discussed the problem of identifying form features from the r-set decomposition of a non-manifold object. In [GTLH98], the authors proposed a decomposition-based technique to convert an arbitrary shape into a manifold one. In [PTL04], the authors proposed a decomposition of a cell 2-complex based on a combinatorial stratification of the complex, inspired by the Whitney stratification [Whi65]. *Selective Geometric Complexes* (SGCs) [RO90] describe objects through cell complexes, encoded through the IG data structure, whose cells can be either open, or not connected. In [LT97], the authors defined a stratification of manifold cell 3-complexes, as well update operators. In [DFMMP03], a decomposition of an arbitrary shape into nearly manifold components has been defined for the purpose of defining a compact data structure for non-manifold shapes discretized as simplicial complexes in arbitrary dimensions. The components are regular d-complexes such that the star of each vertex is $(d-1)$-connected.

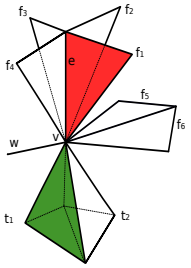
4. The IA* Data Structure

The *Generalized Indexed data structure with Adjacencies* (IA*) [CDFW11] is a dimension-independent data structure that encodes an arbitrary shape, discretized as a simplicial d-complex Σ. The IA* data structure encodes all top simplices in Σ, plus the following topological relations:

- the boundary relation $\mathcal{R}_{p,0}(\sigma)$, for each top p-simplex σ, with $0 < p \leq d$;
- the partial adjacency relation $\mathcal{R}^*_{p,p}(\sigma)$, restricted to top p-simplices adjacent to a top p-simplex σ, with $p > 1$;
- the partial co-boundary relation $\mathcal{R}^*_{p-1,p}(\tau)$, consisting of all top p-simplices incident at a $(p-1)$-simplex τ, $0 < p-1 < d$, on the boundary of more than two top p-simplices;
- the partial co-boundary relation $\mathcal{R}^*_{0,1}(v)$, consisting of all wire-edges incident at a vertex v;
- the partial co-boundary relation $\mathcal{R}^*_{0,p}(v)$, with $p \leq 2$, consisting of one arbitrarily selected top p-simplex for each $(p-2)$-connected component in $Lk(v)$.

Relation $\mathcal{R}^*_{p-1,p}(\tau)$ allows for an efficient encoding for the adjacency relation $\mathcal{R}^*_{p,p}(\sigma)$ of a top p-simplex σ along one of its $(p-1)$-faces τ. It can be encoded as either a top p-simplex adjacent to σ along τ, or as a pointer to the list of all top p-simplices incident at τ. Again, relation $\mathcal{R}^*_{p-1,p}(\tau)$ allows detecting non-manifold simplices τ in constant time. Recall that a $(d-1)$-simplex is non-manifold if it is shared by more than two d-simplices. Hence, a $(d-1)$-simplex τ is non-manifold if $\mathcal{R}^*_{d-1,d}(\tau) \neq \emptyset$. Figure 2 illustrates the relations encoded by the IA* data structure with a simplicial 3-complex. Clearly, the IA* data structure, for manifolds, reduces to the IA data structure, [PBCF93], since there is not any top simplex of dimension different than d.

The IA* data structure encodes only vertices and top simplices in Σ. They can be referenced through a pair (h,i), called *SimplexPointer*, where h and i are the dimension and the unique identifier in Σ of an encoded h-simplex, respectively. We call non-top simplices *ghost simplices*, and implicitly refer them as the p-face of a top h-simplex in Σ, with $p \leq h$. Thus, a ghost simplex can be referenced through a tuple (h,i,p,j), called the *GhostSimplexPointer*, i.e. the j-th face of dimension p in the top simplex (h,i).



$$\mathcal{R}^*_{0,1}(v) = \{w\}$$
$$\mathcal{R}^*_{0,2}(v) = \{f_1, f_5\}$$
$$\mathcal{R}^*_{0,3}(v) = \{t_1\}$$
$$\mathcal{R}^*_{2,2}(f_1) = \mathcal{R}^*_{2,2}(f_2) = \mathcal{R}^*_{1,2}(e)$$
$$\mathcal{R}^*_{2,2}(f_3) = \mathcal{R}^*_{2,2}(f_4) = \mathcal{R}^*_{1,2}(e)$$
$$\mathcal{R}^*_{1,2}(e) = \{f_1, f_2, f_3, f_4\}$$
$$\mathcal{R}^*_{2,2}(f_5) = \{f_6\}$$
$$\mathcal{R}^*_{2,2}(f_6) = \{f_5\}$$
$$\mathcal{R}_{3,3}(t_1) = \{t_2\}$$

Figure 2: *topological relations encoded by the IA* data structure for an arbitrary shape discretized by a simplicial 3-complex.*

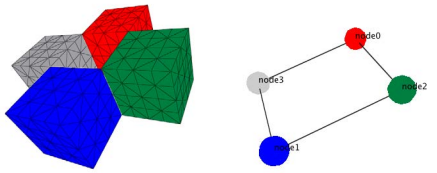
Boundary relations can be retrieved in constant time by exploiting encoded $\mathcal{R}_{d,0}$ relations. Top simplices incident at a vertex v can be retrieved by performing a breadth-first visit of all connected components of the link encoded in $\mathcal{R}^*_{0,p}$, thus exploiting partial adjacency relation $\mathcal{R}^*_{p,p}$, with $p > 1$. Time complexity of this operation is linear in the number of top simplices incident at v. Co-boundary relation of a generic simplex σ can be retrieved in two steps: first, we retrieve all simplices incident at one of its vertices, and then we retrieve the subset of such simplices incident in all vertices of σ. Time complexity is dominated by the retrieval of all top simplices incident at a vertex v of σ, thus it is linear in the number of top simplices incident at v. Adjacency relations $\mathcal{R}^*_{p,p}$, restricted to top p-simplices, with $p > 1$, are already encoded. The partial adjacency relation $\mathcal{R}^*_{1,1}(\sigma)$ can be easily retrieved through encoded relations $\mathcal{R}^*_{0,1}$ and $\mathcal{R}_{1,0}$. Generally speaking, the adjacency relation for a k-simplex can be retrieved by combining relations $\mathcal{R}_{k,k-1}$ and $\mathcal{R}_{k,k+1}$.

5. The Manifold-Connected Decomposition

5.1. Manifold-connected components

Recall that a $(d-1)$-simplex σ in a regular simplicial d-complex Σ is a *manifold* simplex if and only if there exists at most two d-simplices in Σ incident at τ. Two d-simplices σ' and σ'' are *manifold-connected* if and only if there exists a $(d-1)$-path joining σ' and σ'' such that any two consecutive d-simplices are adjacent through a manifold $(d-1)$-simplex. A regular d-complex in which every pair of simplices is manifold-connected is a *manifold-connected (MC) complex*. Note that any combinatorial manifold is manifold-connected, but the reverse is not true. Thus, the class of MC complexes is a decidable superset of combinatorial manifolds. Note that the class of d-manifolds is not decidable for $d \geq 6$, [DFMMP03]. Up to dimension 2, the class of MC complexes coincides with manifolds. MC 3-complexes, if embedded in $\mathbb{E}^3$, are pseudo-manifolds.

A decomposition of an arbitrary shape into simpler parts can be obtained by splitting it along non-manifold singularities and without introducing artificial "cuts" in the manifold parts. Our analysis starts from regular simplicial complexes.The manifold-connectivity relation between the top d-simplices in a regular d-complex Σ defines an equivalence relation. The MC-components of Σ are the equivalence classes of the top d-simplices in Σ within respect to the manifold-connectivity relation. Hence, the collection of all MC-components in Σ forms the *Manifold-Connected Decomposition* (MC-decomposition) of Σ. Any component in the MC-decomposition of Σ may have a common intersection which is a sub-complex Σ of dimension lower than d. Figure 3(a) shows the MC-decomposition of a regular simplicial 2-complex, where the four MC-components are connected through chains of non-manifold edges.



(a) (b)

Figure 3: *(a) MC-components and (b) the corresponding Pair-wise MC-graph of the* cubes *shape. Each MC-component is identified by its center of gravity in the* CCViewer *program, [DFPH09].*

We describe how to compute the MC-decomposition of an arbitrary simplicial d-complex Σ. Note that the algorithm is entirely dimension-independent. We notice that a simplicial d-complex Σ is uniquely decomposed into a collection of maximal regular complexes Σ_k formed by top k-simplices, with $k \leq d$. Thus, the MC-decomposition of Σ is the collection of the MC-decompositions of the maximal regular sub-complexes Σ_k.

The computation of the MC-decomposition of Σ starts by cutting it into its regular sub-complexes Σ_k. Then, for each sub-complex Σ_k, we perform a traversal of Σ_k, starting from a not visited top k-simplex σ, and retrieve all top k-simplices in Σ_k that are reachable from σ by visiting manifold $(k-1)$-simplices and their incident top k-simplices. All top k-simplices reachable from σ belong to the same MC-component. Each MC-component is assigned a unique label $\mathcal{C}$, and every simplex in this component, including singularities, is assigned $\mathcal{C}$. As a consequence, we can also retrieve a list of singularities in Σ. Thus, it must be possible to extract the boundary, the star, and the adjacency relation of any simplex in Σ, and to identify a non-manifold singularity in Σ. We encode Σ through an instance of the IA* data structure. Algorithm 1 provides a pseudo-code description of this traversal. At the end of this algorithm, array L has a number of elements equal to the number of singularities τ in Σ, and each location $L[\tau]$ contains the list of MC-components incident at τ. The traversal of each sub-complex Σ_k is a process linear in the number of top k-simplices in Σ_k. At each step, we analyze all faces of each top k-simplex, looking for

non-manifold singularities. Generally speaking, these simplices are not encoded in the IA* data structure, and thus they are implicitly represented through their *GhostSimplexPointer*. An arbitrary shape Σ is uniquely decomposed into the sub-complexes Σ_k, thus the time complexity is linear in the number $\mathcal{S}_\Sigma$ of simplices in Σ.

Algorithm 1 RETRIEVE_MC_COMPONENTS(Σ)

Input: an instance of the IA* data structure representing Σ
Output: the set of non-manifold singularities in Σ and their related MC-components

```
1: let L := ∅
2: for all k = 1,...,dim(Σ) do
3:   for all top k-simplex σ in Σ do
4:     if σ is not visited then
5:       create a new MC-component C
6:       let q an empty queue
7:       enqueue σ in q
8:       while q is not empty do
9:         dequeue σ' from q
10:        if σ' is not visited then
11:          mark σ' as visited
12:          add σ' in the new MC-component C
13:          for all τ in b(σ') do
14:            if τ is not manifold in Σ then
15:              L[τ] := L[τ] ∪ {C}
16:            else if dim(τ) = k − 1 then
17:              enqueue R*_{k,k}(σ') along τ in q
18:            end if
19:          end for
20:        end if
21:      end while
22:    end if
23:  end for
24: end for
25: return L
```

Here, we present two different graph-based data structures, namely the *Extended MC-graph* and the *Pair-wise MC-graph*, for encoding the MC-decomposition of an arbitrary simplicial shape Σ. Both data structures are two-layer representations, in which the upper level describes the connectivity of the MC-components in Σ, while the second layer provides a full description of Σ through an IA* data structure. In this way, MC-components are thus expressed through references to the simplices stored in the second level, similarly as in a spatial index. The Extended MC-graph encodes all singularities in Σ in the connection of the MC-components, and it is suitable to retrieve a *semantic decomposition* of Σ, by combining together into closed components the MC-components that form 2-cycles (shells) in Σ. The Pair-wise MC-graph encodes the intersection of two MC-components, and it is useful, for instance, to iteratively compute the simplicial homology of an arbitrary shape Σ, as discussed in Section 6.

5.2. The Extended MC-graph

The *Extended MC-graph* encodes the MC-decomposition of an arbitrary simplicial shape Σ as a hyper-graph $\mathcal{G}_E = \{\mathcal{N}_E, \mathcal{A}_E\}$, where a node in $\mathcal{N}_E$ corresponds to a MC-component in Σ, and a hyper-arc in $\mathcal{A}_E$ corresponds to a singularity σ in Σ, connecting all MC-components sharing σ. For each node c, we encode:

- the dimension k of the top simplices describing c;
- the s_c references to the top simplices belonging to c;
- the a_c^e references to hyper-arcs incident at c.

Thus, the storage cost of a node c is equal to $(1 + s_c + a_c^e)$ integers. For each arc a in $\mathcal{A}_E$, we encode:

- a *GhostSimplexPointer* reference to the singularity σ related to a;
- the l_σ references to MC nodes incident at σ.

Thus, the storage cost of an arc a is $(4 + l_\sigma)$ integers. Let $\bar{n}$ be the number of MC-components in Σ, s^t be the number of top simplices in Σ, and a_E be the number of hyper-arcs in the Extended MC-graph. Its storage cost is equal to:

$$\bar{n} + s_t + \sum_{c \in \mathcal{N}_E} a_c^e + 4a_E + \sum_{a \in \mathcal{A}_E} l_\sigma \qquad (1)$$

The Extended MC-graph of Σ can be retrieved in two steps. First, we apply Algorithm 1 to completely identify MC-components in Σ. The time complexity of this step is linear in the total number S_Σ of simplices in Σ. As result, we also obtain the array L, which has a number of locations as the number of singularities σ in Σ. Each location $L[\sigma]$ contains l_σ identifiers of all MC-components incident at σ. For each of such locations, we generate a hyper-arc of the Extended MC-graph related to σ. Thus, the time complexity to retrieve the Extended MC-graph is $\mathcal{O}(S_\Sigma)$.

5.3. The Pair-wise MC-graph

The *Pair-wise MC-graph* encodes the MC-decomposition of an arbitrary simplicial shape Σ as a graph $\mathcal{G}_P = \{\mathcal{N}_P, \mathcal{A}_P\}$, where a node in $\mathcal{N}_P$ corresponds to a MC-component in Σ, and an arc in $\mathcal{A}_P$ describes the intersection between two MC-components. For each node c, we encode:

- the dimension k of top simplices describing c;
- the s_c references to the top simplices belonging to c;
- the a_c^p references to arcs incident at c.

Thus, the storage cost of a node c is equal to $(1 + s_c + a_c^p)$ integers. For each arc a in $\mathcal{A}_P$, we encode:

- two references to nodes c_1 and c_2 connected by the arc a;
- l_a^p *GhostSimplexPointer* references to singularities σ belonging to the intersection between the nodes c_1 and c_2.

Thus, the storage cost of an arc is equal to $(4+l_a^p)$ integers. Let a_P be the number of arcs in the Pair-wise MC-graph. The storage cost of the Pair-wise MC-graph is:

$$\bar{n}+s_t+\sum_{c\in\mathcal{N}_P} a_c^p+4a_P+\sum_{a\in\mathcal{A}_E} l_a^p \qquad (2)$$

The Pair-wise MC-graph of an arbitrary shape Σ is retrieved in two steps. First, we apply Algorithm 1 to completely identify MC-components in Σ. The time complexity of this step is linear in the total number S_Σ of simplices in Σ. As a result, we also obtain the array L, which has a number of locations as the number of singularities σ in Σ. Each location $L[\sigma]$ contains l_σ identifiers of all MC-components C_i, with $0 \le i < l_\sigma$, incident at σ. Arcs of the Pair-wise MC-graph are then retrieved as follows:

1. for each non-manifold singularity σ, generate all possible pairs of intersections between the MC-components C_i and C_j, and store tuples (σ, C_i, C_j) in an array $\mathcal{B}$;
2. sort tuples in $\mathcal{B}$ by using the lexicographic order on pairs (C_i, C_j): tuples related to the same pair of labels are stored in consecutive locations of $\mathcal{B}$;
3. create a new arc for each unique pair of nodes identified at step 2, and complete all missing data in the involved nodes and arcs.

Let $\mathcal{M}_\Sigma$ be the list of all non-manifold singularities in Σ, and l_σ be the number of MC-components incident at a singularity σ. Then, the total number of intersections generated at Step 1 is $\bar{l}_\sigma = l_\sigma!/(l_\sigma-2)!$, for each singularity σ. The time complexity is dominated by Step 2, where all elements of $\mathcal{B}$ are sorted: here, the time complexity is in $\mathcal{O}\left(\sum_{\sigma\in\mathcal{M}_\Sigma} \bar{l}_\sigma \log \bar{l}_\sigma\right)$. Thus, the time complexity required to retrieve the Pair-wise MC-graph is $\mathcal{O}\left(\mathcal{S}_\Sigma + \sum_{\sigma\in\mathcal{M}_\Sigma} \bar{l}_\sigma \log \bar{l}_\sigma\right)$.

6. Computing homology of arbitrary shapes

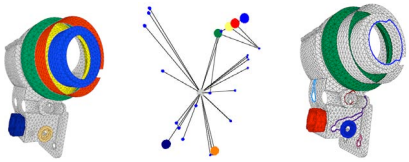
In [BCM*11], we have designed an iterative and modular algorithm for computing the simplicial homology of an arbitrary shape Σ based on the MC-decomposition, that we call *Mayer-Vietoris* (MV) algorithm.

Simplicial homology provides an algebraic representation of simplicial complexes [Mun99]. Each k-simplex σ in a simplicial complex Σ is *oriented* by sorting its vertices, denoted as $\sigma = [v_0,\ldots,v_k]$. The *oriented boundary* of σ is $\delta_k(\sigma) = \sum_{i=0}^{k}(-1)^k[v_0,\ldots,\bar{v_i},\ldots,v_k]$, discarding the vertex v_i at each step. A linear combination of oriented k-simplices is a *k-chain*. The k^{th} *chain-group* $\mathcal{C}_k$ of Σ is formed by all k-chains in Σ together with the addition operation, defined simplex by simplex. An oriented k-simplex is expressed as a $(k-1)$-chain through δ_k, which defines the *boundary homomorphism* $\delta_k : \mathcal{C}_k \to \mathcal{C}_{k-1}$. Coefficients of all $(k-1)$-faces of k-simplices in δ_k can be encoded in the k^{th} *incidence matrix* of Σ. The *chain-complex* $\mathcal{C}_*$ of Σ is an algebraic description of Σ, consisting of the sequence of k-chain groups $\mathcal{C}_k$ related by δ_k, with $k \in \mathbb{N}$. Here, we can recognize the *group of k-cycles*, $\mathcal{Z}_k = \{c \in \mathcal{C}_k.\delta_k(c) = 0\}$, and the *group of k-boundaries*, $\mathcal{B}_k = \{c \in \mathcal{C}_k.\exists a \in \mathcal{C}_{k+1} | c = \lambda_{k+1}(a)$. Clearly, $\mathcal{B}_k \subseteq \mathcal{Z}_k \subseteq \mathcal{C}_k$. Thus, we can define the k^{th} *homology group* as $\mathcal{H}_k = \mathcal{Z}_k|\mathcal{B}_k$, which can be described through its *generators*, one for every equivalence class in $\mathcal{H}_k$. Again, it can be expressed [Mun99] as:

$$\overbrace{\mathbb{Z}\oplus\ldots\oplus\mathbb{Z}}^{\text{free group}} \oplus \overbrace{\mathbb{Z}/\lambda_1\mathbb{Z}\oplus\ldots\oplus\mathbb{Z}/\lambda_p\mathbb{Z}}^{\text{torsion group}}$$

The number of occurrences of $\mathbb{Z}$ in the free part is the k^{th} *Betti number* β_k, while $\lambda_1,\ldots,\lambda_p$ are the *torsion coefficients*. The most common algorithm for computing homology of Σ is the *Smith Normal Form* (SNF) algorithm [Mun99], which reduces all incidence matrices in Σ to their SNFs.

In our approach, we compute the homology of the union of two sub-complexes, starting from their homologies and the homology of their intersection, computed through the SNF algorithm. The starting point of the MV algorithm is the MC-decomposition of an arbitrary shape Σ, encoded through the Pair-wise MC-graph. We have shown that the intersection between two MC-components never exceeds the 5% of S_Σ. Figure 4 shows generators computed by the MV algorithm on the *Carter* shape.



(a) (b) (c)

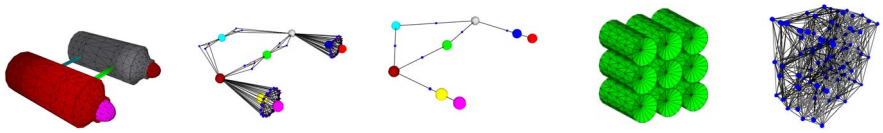
Figure 4: *(a) MC-components, (b) the related Pair-wise MC-graph, and (c) generators computed by the MV algorithm on the* Carter *shape [BCM*11].*

7. Experimental Results

First, we demonstrate that our graph-based data structures are effective tools for representing arbitrary shapes. In Table 1, we compare the storage costs of our graph-based data structures, of the IA* data structure, and of the Incidence Graph, restricted to simplicial complexes. Recall that the IG data structure is the most common data structure for encoding simplicial and cell complexes in arbitrary dimensions. Our results show that, in the large majority of cases, the Extended MC-graph and Pair-wise MC-graph, used in conjunction with the IA* data structure, are more compact than the Incidence Graph. When encoding 2D shapes, on average, the Extended and the Pair-wise MC-graphs (plus the IA*) require about 78% and 86% of the IG data structure respectively, while both these data structures require about 35% of the storage cost of the IG data structure when encoding 3D shapes. Notice that the Incidence Graph is respectively

Shape	n	a_E	a_P	IA*	IG	E	P	E_{IG} (%)	P_{IG} (%)	E_{IA*}(%)	P_{IA*}(%)
Carter (2d)	28	641	40	52.4*k*	95*k*	11.7*k*	9.4*k*	67	65	22	18
Cubes	4	24	4	5*k*	9*k*	948	820	66	65	11	9
Tower-wir	970	2.1*k*	10.7*k*	109*k*	194*k*	31.8*k*	74.3*k*	72.6	94.5	29	68
Pinched pie	120	936	1.4*k*	16.5*k*	25.7*k*	10.8*k*	14.6*k*	106	121	65	88
Chime (3d)	27	29	47	3.2*k*	12*k*	559	653	31	32	17	20
Flasks	8	76	10	32*k*	104*k*	4.4*k*	4.1*k*	35	34.7	14	13
Pinched torus	2	19	1	7*k*	23*k*	948	876	34.5	34	13.5	12.5
Sierpinski	16*k*	32*k*	32*k*	196*k*	917*k*	180*k*	180*k*	41	41	92	92

Table 1: *statistics on our graph-based data structures, encoding a MC-decomposition with* ***n*** *MC-components. Here, we propose storage cost of a shape encoded with the IA* (**IA***) and IG (**IG**) data structures, and the storage cost of the Extended MC-graph (**E**) and Pair-wise MC graph (**P**), restricted to the connectivity of MC-components, which are respectively formed by* $\boldsymbol{a}_E$ *and* $\boldsymbol{a}_P$ *arcs. Columns* $\boldsymbol{E}_{IG}$ *and* $\boldsymbol{P}_{IG}$ *show respectively the complete storage cost of the Extended and Pair-wise MC-graph (including **IA***), expressed as percentage of **IG**. Columns* $\boldsymbol{E}_{IA*}$ *and* $\boldsymbol{P}_{IA*}$ *show respectively **E** and **P** expressed as percentage of **IA***. *These datasets are freely available at [GGG09].*



(a) (b) (c) (d) (e)

Figure 5: *(a) the non-regular* Flasks *3D shape, and the related (b) Extended and (c) Pair-wise MC-graph. (d) The non-regular* Pinched pie *2D shape, and the related (e) Pair-wise MC-graph.*

1.7 and 3.7 times more expensive that the IA* data structure when considering 2D and 3D shapes. We also compare the storage cost required for encoding the connectivity of MC-components with our graph-based data structures. It can be considered as an overhead for the IA* data structure. Our results show that, for both graph-based data structures, this overhead is about 40% (with 2D shapes) and 38% (with 3D shapes) of the storage cost required by the IA* data structure. In any case, it is important to observe that both graph-based data structures are more expressive than the IA* data structure, because they explicitly expose singularities, and the connectivity of the MC-components.

We compare the storage costs of the Extended and the Pair-wise MC-graph, restricted to the connectivity of MC-components. Our results show that the Extended MC-graph is more compact than the Pair-wise MC-graph, in most cases. Specifically, it is about 30% less expensive than the Pair-wise MC-graph with 2D shapes, and about 1% with 3D shapes. However, in some situations, this does not happen, as demonstrated in Table 1, because the complexity of the MC connectivity is reflected in the complexity of the graph-based data structures. For instance, in the *Sierpinski* shape, all MC-components consist of one tetrahedron, sharing a non-manifold vertex with an other tetrahedron. All singularities are non-manifold vertices, and thus the two graph-based data structures coincide. Again, in the non-regular shape *Flasks*, all singularities can be grouped in chains of non-manifold edges describing intersection of only two MC-components, as depicted in Figure 5. Here, the Pair-wise MC-graph becomes about 7% more compact than the Extended MC-graph. Conversely, the Pair-wise MC-graph tends to be strongly-connected if the same set of singularities is shared by more than two MC-components, because we must encode all possible pairs of intersections between MC-components, that may be very relevant. For instance, the Pair-wise MC-graph of the *Pinched pie* shape, depicted in Figure 5, tends to be strongly-connected, since the same chain of non-manifold edges is shared by a large number of MC-components. Also the Extended MC-graph tends to be strongly connected, as confirmed by Table 1. However, in this situation, the Extended MC-graph is about 27% more compact than the Pair-wise MC-graph. Extended and the Pair-wise MC-graphs (plus the IA*) are respectively about 6% and 21% more expensive than the IG data structure.

8. Concluding Remarks

We have addressed the problem of modeling arbitrary shapes discretized as simplicial complexes through a decomposition-based approach. To this aim, we have described a decomposition of such complexes into manifold-connected components, that we call the *Manifold-Connected Decomposition* (MC-decomposition). We have proposed two graph-based data structures, namely the *Extended MC-graph* and the *Pair-wise MC-graph*, for encoding the MC-decomposition. Both data structures are two-layer representations in which the upper level describes the connectivity

of MC-components, while the second layer provides a full description of arbitrary shapes through the IA* data structure. We have performed experimental comparisons and algorithms for generating and navigating such data structures.

In our current and future work, we are developing a semantic-oriented decomposition for 3D shapes discretized through simplicial complexes. This decomposition can be obtained from the MC-decomposition of an arbitrary 3D shape by combining together the MC-components that form a 2-cycle (namely a *shell*). Again, this decomposition can be used as a data structure for representing 3D shapes.

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 contract number IIS-1116747.

References

[BCM*11] BOLTCHEVA D., CANINO D., MERINO S., LEON J.-C., DE FLORIANI L., HETROY F.: An iterative algorithm for homology computation on simplicial shapes. In *Computer-Aided Design* (2011), vol. 43, pp. 1457–1467.

[CDFW11] CANINO D., DE FLORIANI L., WEISS K.: IA*: an adjacency-based representation for non-manifold simplicial shapes in arbitrary dimensions. *Comp. & Graph. - Proc. of SMI '11 35*, 3 (2011), 747–753.

[DFGH04] DE FLORIANI L., GREENFIELDBOYCE D., HUI A.: A data structure for non-manifold simplicial d-complexes. In *Proc. of* 2^{nd} *EG Symp. on Geom. Proc.* 2004, pp. 83–92.

[DFH03] DE FLORIANI L., HUI A.: A scalable data structure for three-dimensional non-manifold objects. In *Proc. of* 1^{st} *EG Symp. on Geom. Proc.* (2003), pp. 72–82.

[DFH05] DE FLORIANI L., HUI A.: Data structures for simplicial complexes: an analysis and comparisons. In *Proc. of* 3^{rd} *EG Symp. on Geom. Proc.* (2005), pp. 119–128.

[DFHPC10] DE FLORIANI L., HUI A., PANOZZO D., CANINO D.: A dimension-independent data structure for simplicial complexes. In *Proc. of Intl. Meshing Rndtbl.* (2010), pp. 403–420.

[DFMMP03] DE FLORIANI L., MESMOUDI M., MORANDO F., PUPPO E.: Non-manifold decompositions in arbitrary dimensions. In *Graph. Models*, no. 65. Elsevier, 2003, pp. 2–22.

[DFMPS04] DE FLORIANI L., MAGILLO P., PUPPO E., SOBRERO D.: A multi-resolution topological representation for non-manifold meshes. *CAD Journal. 36*, 2 (2004), 141–159.

[DFPH09] DE FLORIANI L., PANOZZO D., HUI A.: Computing and visualizing a graph-based decomposition for non-manifold shapes. In *Graph-based Repr. in Patt. Rec.* (2009), pp. 62–71.

[DL89] DOBKIN D., LASZLO M.: Primitives for the manipulation of 3-dimensional subdivisions. *Algor. 5*, 4 (1989), 3–32.

[DS92] DESAULNIER H., STEWART N.: An extension of manifold boundary representation to r-sets. *ACM Trans. on Graphics 11*, 1 (1992), 40–60.

[Ede87] EDELSBRUNNER H.: *Algorithms in Combinatorial Geometry*. Springer, 1987.

[FR92] FALCIDIENO B., RATTO O.: Two-manifold cell decomposition of r-sets. *Comp. Graph. Forum 11* (1992), 391–404.

[GGG09] *Non-manifold Meshes Repository*, 2009. Geometry and Graphics Group, Department of Computer Science, Genoa, Italy, `http://indy.disi.unige.it/nmcollection/`.

[GLLR11] GURUNG T., LANEY D., LINDSTROM P., ROSSIGNAC J.: SQuad: compact representation for triangles meshes. *Comp. Graph. Forum 30*, 2 (2011).

[GR09] GURUNG T., ROSSIGNAC J.: SOT: a compact representation for tetrahedral meshes. In *Proc. of ACM Geom. and Phys. Mod.* (2009), ACM Press, pp. 79–88.

[GTLH98] GUEZIEC A., TAUBIN G., LAZARUS F., HORN W.: Converting sets of polygons to manifold surfaces by cutting and stiching. In *SIGGRAPH Conf.* (1998), ACM Press, pp. 245–265.

[HDF07] HUI A., DE FLORIANI L.: A two-level topological decomposition for non-manifold simplicial shapes. In *Proc. of ACM Geom. and Phys. Mod.* (2007), ACM Press, pp. 355–360.

[LL01] LEE S., LEE K.: Partial-entity structure: a fast and compact non-manifold boundary representation based on partial topological entities. In *Proc. of* 6^{th} *ACM SMA Conf.* (2001), ACM Press, pp. 159–170.

[LLLV05] LAGE M., LEWINER T., LOPES H., VELHO L.: CHF: a scalable topological data structure for tetrahedral meshes. In *Proc. of SIBGRAPI Conf.* (2005), ACM Press, pp. 349–356.

[LT97] LOPES H., TAVARES G.: Structural operators for modeling 3-manifolds. In *Proc. of* 4^{th} *ACM SMA Conf.* (1997), ACM Press, pp. 10–18.

[MMH01] MC-MAINS S., HELLERSTEIN C.: Out-of-core building of a topological data structure from a polygon soup. In *Proc. of* 6^{th} *ACM SMA Conf.* (2001), ACM Press, pp. 171–182.

[Mun99] MUNKRES J.: *Algebraic Topology*. Prentice Hall, 1999.

[PBCF93] PAOLUZZI A., BERNARDINI F., CATTANI C., FERRUCCI V.: Dimension-independent modeling with simplicial complexes. *ACM Trans. on Graphics 12*, 1 (1993), 56–102.

[PTL04] PESCO S., TAVARES G., LOPES H.: A stratification approach for modeling two-dimensional cell complexes. *Comp. & Graphics 28* (2004), 235–247.

[RO90] ROSSIGNAC J., O'CONNOR M.: SGC: a dimension-independent model for point-sets with internal structures and incomplete boundaries. *Geom. Mod. for Prod. Eng.* (1990), 145–180.

[RSS01] ROSSIGNAC J., SAFONOVA A., SZYMCZAK A.: 3d compression made simple: Edgebreaker with zip & wrap on a corner-table. In *Proc. of SMI 2001* (2001), pp. 278–283.

[Wei88] WEILER K.: The radial-edge data structure: a topological representation for non-manifold geometric boundary modeling. *Geom. Mod. for CAD App.* (1988), 3–36.

[Whi65] WHITNEY H.: Local properties of analytic varieties. In *Differential and combinatorial topology*, Cairns S., (Ed.). Princ. Univ. Press, 1965, pp. 205–244.