Eurographics Symposium on Geometry Processing (2004)
R. Scopigno, D. Zorin (Editors)

A data structure for non-manifold simplicial d-complexes

Leila De Floriani[1,2], David Greenfieldboyce[2] and Annie Hui[2]

[1] Department of Computer and Information Sciences,
University of Genova, Via Dodecaneso, 35 - 16146 Genova (Italy).
[2] Department of Computer Science, University of Maryland, College Park, MD 20742 (USA).

Abstract

We propose a data structure for d-dimensional simplicial complexes, that we call the Simplified Incidence Graph (SIG). *The simplified incidence graph encodes all simplices of a simplicial complex together with a set of boundary and partial co-boundary topological relations. It is a dimension-independent data structure in the sense that it can represent objects of arbitrary dimensions. It scales well to the manifold case, i.e. it exhibits a small overhead when applied to simplicial complexes with a manifold domain. Here, we present efficient navigation algorithms for retrieving all topological relations from a SIG, and an algorithm for generating a SIG from a representation of the complex as an incidence graph. Finally, we compare the simplified incidence graph with the incidence graph, with a widely-used data structure for d-dimensional pseudo-manifold simplicial complexes, and with two data structures specific for two- and three-dimensional simplicial complexes.*

1. Introduction

We consider the problem of representing and manipulating non-manifold and non-regular arbitrarily dimensional objects described by simplicial complexes. A *manifold* object is a subset of the Euclidean space for which the neighborhood of each internal point is homeomorphic to an open ball and the neighborhood of each boundary point to an open half ball. Objects, that do not fulfill this property at one or more points, are called *non-manifold* objects. A subset of the d-dimensional Euclidean space containing parts of different dimensionalities is called a *non-regular* d-dimensional object.

Cell complexes are widely used to represent multi-dimensional geometric objects in many applications. In particular, simplicial complexes have received great attention, since their combinatorial properties make them easier to understand, represent and manipulate than more general cell complexes. In the literature, representations have been developed for two-dimensional simplicial and cell complexes describing the boundary of 3D non-manifold and non-regular objects. Several data structures do not scale well with the degree of “non-manifoldness" (i.e., the number of geometric singularities) of the complex, thus becoming often verbose and inefficient when dealing with objects with few non-manifold singularities. Representations developed for cell or simplicial complexes in arbitrary dimensions, (e.g., [2, 15, 17]), are restricted to a subclass of complexes, namely manifold or pseudo-manifold complexes. The incidence graph [9] has been developed for encoding cell complexes in arbitrary dimensions. However, when restricted to simplicial complexes, it results in a verbose representation, which also does not scale well to the manifold case. Scalability is an important property for data structures for cell and simplicial complexes, since non-manifold objects often present few non-manifold singularities.

In this paper, we propose a dimension-independent data structure for d-dimensional simplicial complexes, that we call the *Simplified Incidence Graph (SIG)*. The SIG encodes all simplices of a simplicial complex, the same boundary relations as the incidence graph, but not the complete co-boundary ones, thus exploiting the fact that we are dealing with simplicial complexes. The boundary and partial co-boundary relations encoded in the SIG are the basis for efficient traversal algorithms which retrieve all boundary topological relations in constant time and all co-boundary and adjacency relations in time linear in the number of entities incident at the query entity.

A SIG also scales well to the manifold case, since it requires only a small amount of extra storage compared with a simplified incidence graph specific for manifold simplicial complexes. We will compare the storage cost and the performances of the SIG with those of other data structures, namely, the incidence graph in the d-dimensional case, the indexed data structure with adjacencies for the restricted class of d-dimensional pseudo-complexes complexes for which this latter is defined, and specific compact data structures for 2D and 3D simplicial complexes, which do not extend to higher dimensions.

Novel contributions of this paper are:

- a dimension-independent and scalable data structure, the simplified incidence graph, for representing non-manifold, non-regular, d-dimensional objects described by simplicial complexes;
- efficient navigation algorithms for retrieving adjacency and incidence relations from a simplified incidence graph;
- an algorithm to generate a simplified incidence graph from an incidence graph representation of a complex.

The remainder of this paper is organized as follows. In Section 2, we summarize some background notions. In Section 3, we review some related work. In Section 4, we present the simplified incidence graph, its implementation and its storage cost. In Section 5, we describe an algorithm for generating a SIG from an incidence graph. In Section 6, we present algorithms for retrieving topological relations from a SIG. In Section 7, we compare the simplified incidence graph with other data structures. In Section 8, we draw some concluding remarks.

2. Background

In this section, we review some basic notions about Euclidean simplicial complexes in arbitrary dimensions, and about topological relations among the cells of a simplicial complex.

A Euclidean simplex σ of dimension d is the convex hull of $d+1$ linearly independent points in the n-dimensional Euclidean space E^n, with $d \leq n$. We simply call a *Euclidean d-simplex* a *d-simplex*: a 0-simplex is a *vertex*, a 1-simplex an *edge*, a 2-simplex a *triangle*, and a 3-simplex a *tetrahedron*. d is called the *dimension* of σ and is denoted $dim(\sigma)$. Any Euclidean k-simplex σ', with $k < d$, generated by a set $V_{\sigma'} \subseteq V_\sigma$ of cardinality $k+1 \leq d$, is called a *k-face* of σ. Where no ambiguity arises, the dimensionality of σ' can be omitted and σ' is simply called a *face* of σ. The empty set is a (-1)-face of all simplices.

A finite collection Σ of Euclidean simplices forms a *Euclidean simplicial complex* if and only if (i) for each simplex $\sigma \in \Sigma$, all faces of σ belong to Σ, and (ii) for each pair of simplices σ and σ', either $\sigma \cap \sigma' = \emptyset$ or $\sigma \cap \sigma'$ is a face of both σ and σ'. The *domain*, or *carrier*, of a Euclidean simplicial complex Σ embedded in E^n, with $d \leq n$, is the subset of E^n defined by the union, as point sets, of all the simplices in Σ.

The *boundary* $b(\sigma)$ of a simplex σ is the set of all faces of σ. The *co-boundary*, or *star*, of a simplex σ is defined as $\star\sigma = \{\xi \in \Sigma \mid \sigma \text{ is a face of } \xi\}$. A simplex σ is called a *top simplex* of Σ if $\star\sigma = \{\sigma\}$ In the following, we will call *restricted star* of a simplex σ, $\star\sigma - \{\sigma\}$, and we will denote it as $st(\sigma)$.

Two distinct simplices are said to be *incident* if one of them is a face of the other. Two simplices are called *k-adjacent* if they share a k-face. Two p-simplices, with $p > 0$, are said to be *adjacent* if they are $(p-1)$-adjacent. Two vertices (i.e., 0-simplices) are called *adjacent* if they are both incident at a common 1-simplex.

An *h-path* is a sequence of simplices $(\sigma_i)_{i=0}^k$ such that two consecutive simplices σ_{i-1} and σ_i in the sequence are h-adjacent. Two simplices σ and σ^* are *h-connected* if and only if 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 complex Σ is called *h-connected* if and only if any two simplices of Σ^* are h-connected. Any maximal h-connected sub-complex of a complex Σ is called an *h-connected component* of Σ. We call any $(h-1)$-connected component in which all top simplices have dimension h an *h-cluster*. For example, a 2-cluster is a set of edge-adjacent triangles.

A d-complex Σ in which all top simplices are d-simplices is called *regular*, or *uniformly d-dimensional*. A regular $(d-1)$-connected d-complex in which the star of all $(d-1)$-simplices consists of one or two simplices is called a *(combinatorial) pseudo-manifold* (possibly with boundary).

Let Σ be a d-complex and let $\sigma \in \Sigma$ be a p-simplex, with $0 \leq p \leq d$. For g,q, $0 \leq g,q \leq d$, we define the following *topological relations*:

- For $p > q$, the *boundary relation* $B_{p,q}(\sigma)$ consists of the set of simplices of order q in the set of faces of σ.
- For $p < g$, the *co-boundary relation* $C_{p,g}(\sigma)$ consists of the set of simplices of order g in the star of σ.
- For $p > 0$, the *adjacency relation* $A_p(\sigma)$ is the set of p-simplices in Σ that are $(p-1)$-adjacent to σ.
- The *adjacency relation* $A_0(\sigma)$, where σ is a vertex, consists of the set of vertices σ' such that $\{\sigma, \sigma'\}$ is a 1-simplex of Σ.

Boundary and co-boundary relations are called *incidence relations*.

To describe the g-clusters incident at a simplex σ, we define the following *partial co-boundary relation*:

- For $p < g$, $C^*_{p,g}(\sigma)$ consists of one arbitrarily-selected g-simplex for each g-cluster in the restricted star $st(\sigma)$ of σ

In particular, relation $C^*_{p,p+1}(\sigma)$ consists of all top $(p+1)$-simplices in the restricted star $st(\sigma)$ of σ, since each $(p+1)$-cluster consists of just one top $(p+1)$-simplex.

3. Related Work

Dimension-independent data structures have been proposed for d-dimensional manifold complexes, which include the *cell-tuple* data structure [2], the *n-G-map* [15] for cell complexes, and the *Indexed data structure with Adjacencies (IA)* for simplicial complexes (also called *winged representation* [17]). In the IA data structure, $2(d+1)$ references are needed for each d-simplex in a d-complex. Both the cell-tuple data structure, and the n-G-map, when used to describe a simplicial complex, require $(d+1)!(d+1)$ references for each d-simplex. The IA can describe Euclidean pseudo-manifolds embedded in the d-dimensional space. The n-G-map and the cell-tuple data structures describe a larger subclass of pseudo-manifolds introduced in [15]. However, none of them can encode arbitrary cell or simplicial complexes.

Selective Geometric Complexes (SGCs) [19] can describe non-manifold and non-regular objects through cell complexes whose cells can be either open, or not simply connected. In SGCs, cells and their mutual adjacencies are encoded in an incidence graph [9]. The *incidence graph* is a data structure for arbitrary cell complexes, which encodes all the cells of the complex, and for each k-cell γ, all $(k-1)$-cells bounding γ, and all $(k+1)$-cells in the restricted star of γ. Thus, it provides a complete, but verbose description of the complex.

Data structures for non-manifold, non-regular two-dimensional cell complexes have been proposed for modeling non-manifold solids [12, 13, 20]: any three-dimensional cell is encoded implicitly through the manifold 2-complex partitioning its boundary. Experimental evaluations reported in [14] show that these data structures do not scale well to the manifold case, since their storage cost is between 2.1 and 4.4 times higher than that of the *winged edge* data structure [1]. The *partial entity structure* [14] has been shown to require half of the space of the radial-edge structure. A data structure for encoding two-dimensional simplicial complexes, called the *triangle-segment* data structure, has been proposed in [5], which extends the IA data structure to deal with non-regular parts and with non-manifold adjacencies of two-simplices at an edge. The triangle-segment data structure is compact, and scales very well to the manifold case, since it requires just one byte per vertex more than the IA data structure when applied to a manifold complex.

Two representations have been proposed in the literature for three-dimensional manifold complexes, i.e., the *facet-edge* [8] and the *handle-face* data structures [16]. Both of them describe three-dimensional cells implicitly by encoding the manifold complexes that form their boundary. In [4], a compact data structure for arbitrary three-dimensional simplicial complexes embedded in the three-dimensional Euclidean space is described, called the *Non-Manifold Indexed data structure with Adjacencies (NMIA)*. The NMIA data structure scales well to the case of simplicial three-dimensional manifold complexes, since it exhibits an overhead of just one byte per vertex with respect to the IA data structure when applied to manifold complexes.

An alternative approach to the design of non-manifold data structures consists of decomposing a non-manifold object into simpler and more manageable parts [7, 10, 11, 18]. Such techniques deal with the decomposition of the boundary of a regular object into two-manifold parts. In [6], a sound decomposition for d-dimensional non-manifold objects described through simplicial complexes is defined, which is unique and produces a description of a d-complex as a combination of nearly manifold components.

4. The Simplified Incidence Graph

4.1. Design of the Data Structure

The *Simplified Incidence Graph (SIG)* is a representation for a d-dimensional Euclidean simplicial complex embedded in n-dimensional Euclidean space, with $d \leq n$. When $d = n$, every $(d-1)$-simplex is shared by at most two d-simplices, since any d-dimensional simplicial complex embedded in the d-dimensional Euclidean space is a pseudo-manifold.

Given a d-dimensional simplicial complex Σ, the SIG encodes all p-simplices for $p = 0, 1, \ldots d$ in Σ, and

- for each p-simplex σ, where $0 < p \leq d$, it encodes boundary relations $B_{p,p-1}(\sigma)$,
- for each p-simplex σ, where $0 \leq p < d$, partial co-boundary relations $C^*_{p,g}(\sigma)$ (where $g > p$)

Note that partial co-boundary relation $C^*_{d-1,d}(\sigma)$ is the same as co-boundary relation $C_{d-1,d}(\sigma)$. Moreover, when the domain is a manifold, all partial co-boundary relations are empty with the exception of $C^*_{p,d}(\sigma)$. In this case, relation $C^*_{p,d}(\sigma)$ encodes one or two d-simplices incident at σ when $p = d-1$, or just one d-simplex incident at σ when $p < d-1$.

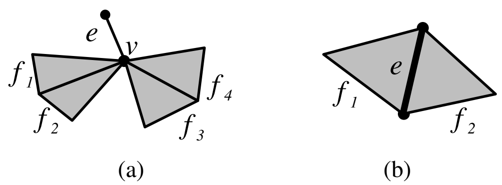
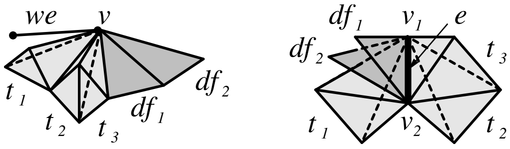


Figure 1: *Two examples showing the relations stored at a vertex (a) and at an edge (b) in a SIG describing a two-dimensional complex. In (a), the top simplices incident at vertex v are triangles f_1, f_2, f_3 and f_4, and edge e. In (b), the top simplices incdient at edge e are triangles f_1 and f_2.*

Figure 1(a) shows an example of the encoding of a vertex of a 2-complex in a SIG. Two partial co-boundary relations are defined at v, namely, $C^*_{0,g}(v)$, for $g = 1, 2$. Relation $C^*_{0,1}(v) = \{e\}$ and relation $C^*_{0,2}(v) = \{f_1, f_2\}$. Figure

1(b) shows an example of the encoding of an edge of a 2-complex in a SIG. The partial co-boundary relation defined at e is $C^*_{0,1}(e)$, which consists of $\{f_1, f_2\}$.



(a) (b)

Figure 2: *Two examples showing the relations stored at a vertex (a) and at an edge (b) in a SIG describing a three-dimensional complex. In (a), the top simplices incidnet at vertex v are tetrahedra t_1, t_2 and t_3, triangles df_1 and df_2, and edge we. In (b), the top simplices incident at edge $e = \{v_1, v_2\}$ are tetrahedra t_1, t_2 and t_3, and triangles df_1 and df_2.*

Figure 2(a) shows an example of the encoding of a vertex of a 3-complex in a SIG. The restricted star of v $st(v)$ consists of edge we, of triangles df_1 and df_2, of the edges of df_1 and df_2 which are incident at v, of tetrahedra t_1, t_2 and t_3, together with all the faces and edges of t_1, t_2 and t_3 which are incident at v. Three partial co-boundary relations are defined at v, namely, $C^*_{0,g}(v)$, for $g = 1,2,3$. Relation $C^*_{0,1}(v) = \{we\}$, relation $C^*_{0,2}(v) = \{df_1\}$, and relation $C^*_{0,3}(v) = \{t_1, t_2\}$.

Figure 2(b) shows an example of the encoding of an edge of a 3-complex in the SIG. The restricted star $st(e)$ of edge e is composed of triangles df_1 and df_2, of tetrahedra t_1, t_2, and t_3, and their faces which are incident at e. Since $e \notin st(e)$, df_1 and df_2 are not 1-connected in $st(e)$, and, thus, they form two separate 1-clusters in $st(e)$. Boundary relation $B^*_{1,0}(e)$, and the two partial co-boundary relations $C^*_{1,g}(e)$, for $g = 2,3$, are stored at edge e. Boundary relation $B^*_{1,0}(e)$ consists of the set of extreme vertices of e, namely $\{v_1, v_2\}$. Partial co-boundary relations $C^*_{1,2}(e)$ and $C^*_{1,3}(e)$ consist of $\{df_1, df_2\}$ and $\{t_1, t_2\}$, respectively.

4.2. Implementation and Storage Cost

In this subsection, we describe the implementation of the SIG, and discuss the storage cost for this implementation. For simplicity, in our current implementation, we use one integer to index a simplex.

All simplices are stored in ascending order of their dimensions. Each simplex has a unique index. A lookup-table is used to encode the starting and ending indices of the simplices for each dimension. For each simplex, we also store a one-bit flag that is used by the navigation algorithms to mark a simplex as visited during traversal, and reset after traversal is completed.

For each p-simplex σ, with $0 < p \leq d$, boundary relation $B_{p,p-1}(\sigma)$ is stored in a fix-sized array, each element of which is an index to a simplex on the boundary of σ. For each p-simplex σ, with $0 \leq pr < d$, partial co-boundary relations $C^*_{p,g}(\sigma)$ for $p < g \leq d$ are stored, in decreasing order of g, in a variable-sized array. Each entry of the array consists of the index of a simplex containing σ in its boundary. The end of the array is marked by a stop code. An integer pointer is associated with simplex σ, which is the starting index of the variable-sized arrays of the co-boundary relations. In the manifold case, for all p-simplices σ, with $p < d-1$, only relation $C^*_{p,d}(\sigma)$ exists. Thus, the integer pointer associated with σ references directly the d-simplex in $C^*_{p,d}(\sigma)$ relation. A flag is used to indicate whether the manifold condition holds at a p-simplex, when $p < d-1$.

We denote with n_p, for $0 \leq p \leq d$, the number of p-simplices in a simplicial complex Σ, with $\kappa^g(\sigma)$, for $dim(\sigma) < g \leq d$, the total number of g-clusters in the restricted star of a simplex σ in Σ, and with $\kappa^g_p = \sum_{dim(\sigma)=p} \kappa^g(\sigma)$, for $0 \leq p < d$, and $g > p$, the total number of g-clusters summed over the restricted stars of all the p-simplices in Σ.

The lookup-table, that stores the starting and ending indices for the simplices in each dimension, requires $d+1$ integers. A total of $\sum_{0 \leq p \leq d} n_p$ bits is needed for the flag used for navigation. The space use for all boundary relations $B_{p,p-1}$ for $0 < p \leq d$ is equal to $\sum_{0<p \leq d} (p+1) n_p$ integers.

The storage space required for encoding the partial co-boundary relations can be evaluated as follows. Each p-simplex, $0 \leq p < d$, has a link to the partial co-boundary relations $C^*_{p,g}$ associated with it. This requires $\sum_{0 \leq p < d} n_p$ integers in total. In addition, the flag, that indicates whether the manifold condition holds at a simplex, requires one bit for each simplex, and, thus, $\sum_{0 \leq p < d} n_p$ bits in total. The total space use for all the variable-sized arrays that store the partial co-boundary relations $C^*_{p,g}$ is equal to $\sum_{0 \leq p < d} \sum_{p < g \leq d} \kappa^g_p + \sum_{0 \leq p < d} n_p$ integers, where the first term accounts for the indices of the simplices and the second term $\sum_{0 \leq p < d} n_p$ accounts for the stop codes.

Thus, the total space used for encoding all topological relations (i.e., both boundary and partial co-boundary relations) is equal to $\sum_{0 \leq p < d} \sum_{p < g \leq d} \kappa^g_p + 2 \sum_{0 \leq p < d} n_p + \sum_{0 < p \leq d} (p+1) n_p$ integers and $\sum_{0 \leq p < d} n_p$ bits.

If Σ is a manifold complex, the variable-sized arrays are not used for the partial co-boundary relations associated with the p-simplices, when $0 \leq p < d-1$. For the $(d-1)$-simplices, the variable-sized arrays are still used because each $(d-1)$-dimensional simplex may be shared by either one or two d-simplices. The space used for encoding the partial co-boundary relations thus becomes $\sum_{0 \leq p < d} n_p + \kappa^d_{d-1} + n_{d-1}$ integers and $\sum_{0 \leq p < d} n_p$ bits. Also, $\kappa^d_{d-1} = 2n_{d-1}$. The space used for the simplices and the boundary relations is the same as in the non-manifold case. Thus, the overhead with respect to a simplified incidence graph specific for a d-complex with a manifold domain is equal to $\sum_{0 \le p<d} n_p$ bits + n_{d-1} integers.

5. Generating a Simplified Incidence Graph

In this section, we present an algorithm for generating the simplified incidence graph from the incidence graph. Given a d-dimensional simplicial complex Σ, the *Incidence Graph (IG)* encodes all the p-simplices σ, $0 \le p \le d$. Also, for each p-simplex, σ, for $0 \le p < d$, it encodes co-boundary relation $C_{p,p+1}(\sigma)$, and, for each p-simplex σ, $0 < p \le d$, boundary relation $B_{p,p-1}(\sigma)$.

Boundary relations $B_{p,p-1}(\sigma)$, for $0 < p \le d$, in the SIG can be directly obtained from the incidence graph. To build partial co-boundary relations $C^*_{p,g}(\sigma)$ for $g > p$, the set S of top simplices that are incident at σ are retrieved from the IG. For each g-simplex σ' in S, boundary relation $B_{g,g-1}(\sigma')$ and co-boundary relation $C_{g-1,g}(\sigma')$ are used in combination to find all the simplices in S that are $(g-1)$-adjacent to σ', i.e., which form a g-cluster with σ. For each g-cluster in S, one simplex is arbitrarily selected to form the partial co-boundary relation $C^*_{p,g}(\sigma)$.

Algorithm **BuildSIGFromIG** provides a pseudo-code description of the technique for constructing all the entities, boundary relations and partial co-boundary relations in a SIG G from an IG I. It makes use of two auxiliary procedures: **RetrieveTopSimplexes** and **VisitCluster**. **RetrieveTopSimplexes** retrieves, from an IG I, all top simplices incident at a simplex σ. Given an r-simplex σ in a set of simplices S, **VisitCluster** visits all the simplices that form one r-cluster with σ.

Algorithm 1 : BuildSIGFromIG(*I*, *G*)
given an Incidence Graph, I, it constructs all the entities, boundary relations and partial co-boundary relations for the Simplified Incidence Graph, G.

```
BuildSIGFromIG(I, G)
Input : an Incidence Graph, I
Output : a Simplified Incidence Graph, G

create all entities σ in G

{define the boundary relations in G}
for all σ ∈ I do
    let p := dim(σ)
    if p > 0 then
        define B_{p,p-1}(σ) in G to be the same as that in I
    end if
end for

{define the partial co-boundary relations in G}
for all σ ∈ I do
    let p := dim(σ), and C*_{p,g}(σ) := ∅ for g > p
    S :=RetrieveTopSimplexes(I, σ)
    while there exists σ' ∈ S, σ' ≠ σ and
        not visited(S, σ') do
        let r := dim(σ')
        C*_{p,r}(σ) := C*_{p,r}(σ) ∪ {σ'}
        if r = p + 1 then
            mark σ' as visited in S
        else
            S := VisitCluster(S, σ')
        end if
    end while
end for
```

6. Retrieving Topological Relations

In this section, we present navigation algorithms to retrieve all incidence and adjacency topological relations from a simplified incidence graph.

6.1. Retrieving Boundary Relations

Boundary relation $B_{p,q}(\sigma)$, with $q < p$, consists of all q-faces of p-simplex σ. It is retrieved as follows. Given a p-simplex σ and $q < p$, we examine $B_{r,r-1}$ relation for each of the r-faces of σ, $r = p, \cdots, q+1$. Algorithm **Boundary** provides a pseudo-code description of such technique.

The time complexity $\mathcal{B}^p_q$ of the algorithm for retrieving boundary relation $B_{p,q}(\sigma)$ is thus $\mathcal{B}^p_q = c\Pi_{r=q+1,p+1} r$, which is bounded by a constant which depends on the dimension p of the simplex and on the dimension q of its faces. For instance, for $B_{d,0}$, $\mathcal{B}^d_0$ is $O((d+1)!)$.

6.2. Retrieving Co-boundary Relations

Co-boundary relations $C_{p,g}(\sigma)$ for a p-simplex σ and $g > p$ are retrieved in two steps. The first step consists of retrieving all top simplices of dimension $\ge g$ in the restricted star $st(\sigma)$ of σ. The second step consists of retrieving, from each of these top simplices, their g-faces that are in $st(\sigma)$.

In the first step, for each r-simplex μ that belongs to $C^*_{p,r}(\sigma)$, for $g \le r \le d$ and $g > p+1$, we retrieve all the r-simplices, that are in the same r-cluster as μ and are in $st(\sigma)$. To this aim, we examine the $(r-1)$-faces of μ in boundary relation $B_{r,r-1}(\mu)$ to see whether they contain σ as a p-face. For each of the $(r-1)$-faces μ', that is incident at σ, all top simplices μ'' in relation $C^*_{r-1,r}(\mu')$ are retrieved. The same procedure is applied to each of these simplices μ'' until all top simplices in the same r-cluster as μ in $st(\sigma)$ are visited. When $g = p+1$, each $(p+1)$-simplex μ in relation $C^*_{p,p+1}(\sigma)$ forms a cluster by itself. Algorithm **RetrieveCluster** provides a pseudo-code description. Given a simplified inci-

Algorithm 2 :RetrieveTopSimplexes(I, σ)
it retrieves from an Incidence Graph, I, the set of top simplices, S, that are incident at a simplex σ.

RetrieveTopSimplexes(I, σ)
Input : an Incidence Graph, I,
a simplex, σ
Output : a set of top simplices, S, incident at σ if σ is not a top simplex, otherwise, a set containing σ

```
let p := dim(σ), S := ∅
if not visited(I, σ) then
    mark σ as visited in I
    if C_{p,p+1}(σ) = ∅ then
        S := {σ}
    else
        for all σ' in C_{p,p+1}(σ) do
            S := S ∪ RetrieveTopSimplexes(I, σ')
        end for
    end if
end if
return S
```

Algorithm 3 : VisitCluster(S, σ)
given a set S of simplices and a r-simplex $\sigma \in S$, it visits the simplices in S, that are $(r-1)$-adjacent to σ.

VisitCluster(S, σ)
Input : a set of top simplices, S,
a simplex, σ, of dimension r in S
Output : a set of top simplices, S, in which simplices that are $(r-1)$-adjacent to σ are marked as visited

```
let r := dim(σ)
if not visited(S, σ) then
    mark σ as visited in S
    for all σ' in B_{r,r-1}(σ) do
        for all σ'' in C_{r-1,r}(σ') do
            if σ'' ∈ S then
                S := VisitCluster(S, σ'')
            end if
        end for
    end for
end if
return S
```

dence graph G, a p-simplex σ, and an r-simplex μ incident at σ, **RetrieveCluster** retrieves all top simplices in $st(\sigma)$ belonging to the same r-cluster as μ.

The second step consists of retrieving, for each top r-simplex μ incident at σ, the g-faces of μ that are incident at σ. When μ is of dimension greater than g, the g-faces θ of μ are obtained from boundary relation $B_{r,g}(\mu)$. Boundary relation $B_{g,p}(\theta)$ is examined for each θ to check whether θ

Algorithm 4 : Boundary(G, q, σ)
it retrieves the q-simplices on the boundary of p-simplex σ.

Boundary(G, q, σ)
Input : a Simplified Incidence Graph, G,
a dimension, $q < p$,
a simplex, σ
Output : a set of q-simplices, S, on the boundary of σ

```
let p := dim(σ), S := ∅
for all σ' ∈ B_{p,p-1}(σ) do
    if not visited(G, σ') then
        mark σ' as visited in G
        if p = q+1 then
            S := S ∪ {σ'}
        else
            S := S ∪ Boundary(G, q, σ')
        end if
    end if
end for
return S
```

is incident at σ. Algorithm **Coboundary** provides a pseudo-code description of the two-step technique for retrieving a co-boundary relation.

To analyze the complexity of the algorithm, given a p-simplex σ, we define $\lambda^r(\sigma)$, for $p < r \leq d$, to be the number of top simplices of dimension r in the restricted star $st(\sigma)$ of σ. In the first step, algorithm **RetrieveCluster** visits its every r-simplex in $st(\sigma)$ exactly once. For each $r+1$ $(r-1)$-faces of each top r-simplex, with $r > p+1$, boundary relation $B_{r-1,p}$ is retrieved to test whether p-simplex σ belongs to $B_{r-1,p}$. Recall that the worst-case time complexity of retrieving boundary relation $B_{r-1,p}$ is equal to $\mathcal{B}_p^{r-1}$. Thus, the complexity of algorithm **RetrieveCluster** is $O(\lambda^g(\sigma)+\sum_{g<r\leq d} r\mathcal{B}_p^{r-1}\lambda^r(\sigma))$.

To analyze the complexity of the second step, we denote with $\mathcal{K}_g^r$ the number of g-faces of an r-simplex. The worst-case time complexity for retrieving the g-faces of an r-simplex is $\mathcal{B}_g^r$. Testing whether a g-face is incident at σ takes $\mathcal{B}_p^g$ operations. In total, the second step requires $O(\sum_{g<r\leq d}\lambda^r(\sigma)(\mathcal{B}_g^r+\mathcal{K}_p^r\mathcal{B}_p^g))$ operations.

Let $\mathcal{C}_p^g$ denote the number of operations performed by algorithm **Coboundary**. Then $\mathcal{C}_p^g = O(\lambda^g(\sigma) + \sum_{g<r\leq d}\lambda^r(\sigma)(\mathcal{B}_g^r + \mathcal{K}_p^r\mathcal{B}_p^g + r\mathcal{B}_p^{r-1}))$ in the worst case. $\mathcal{C}_p^g$ is linear in the number of top simplices of dimension from $p+1$ to g incident at p-simplex σ. The multipliers ($\mathcal{B}_g^r$, $\mathcal{K}_p^r\mathcal{B}_p^g$ and $r\mathcal{B}_p^{r-1}$) for the terms $\lambda^r(\sigma)$, for $g < r \leq d$, are bounded by constants that depend only on the dimensions p and g.

For instance, retrieving $C_{0,d}(v)$ relation for a vertex v requires $O(\lambda^d(v))$ time, which is equal to the number of d-

simplices incident at v, regardless of whether the manifold condition holds at v.

Algorithm 5 : Coboundary(G, g, σ)
it retrieves the g-simplices in the restricted star of p-simplex σ.

Coboundary(G, g, σ)
Input : a Simplified Incidence Graph, G,
 a dimension, $g > p$,
 a simplex, σ
Output : a set of simplices, T, in the restricted star of σ

```
let p := dim(σ), T := ∅
for all σ′ ∈ C*_{p,r}(σ) for g ≤ r ≤ d do
    {Step 1:}
    S := RetrieveCluster(G, σ, σ′)
    {Step 2:}
    if r = g then
        T := T ∪ S
    else
        for all μ ∈ S do
            S′ := Boundary(G, g, μ)
            for all θ ∈ S′ do
                if σ ∈ Boundary(G, p, θ) then
                    T := T ∪ {θ}
                end if
            end for
        end for
    end if
end for
return T
```

Algorithm 6 : RetrieveCluster(G, σ, μ)
given a simplified incidence graph, G, a p-simplex σ, and an r-simplex μ in the restricted star of σ, $st(\sigma)$, it retrieves all the simplices that are $(r-1)$-connected to μ in $st(\sigma)$.

RetrieveCluster(G, σ, μ)
Input : a Simplified Incidence Graph, G,
 a simplex, σ, of dimension p,
 a simplex, μ, of dimension $r > p$, incident at σ
Output : a set of simplices in the same r-cluster as μ, S

```
let p := dim(σ), r := dim(μ), and S := ∅
if not visited(G, μ) then
    mark μ as visited in G
    S := S ∪ {μ}
    if r > p+1 then
        for all μ′ ∈ B_{r,r−1}(μ) do
            if σ ∈ Boundary(G, p, μ′) then
                for all μ″ ∈ C*_{r−1,r}(μ′) do
                    S := S ∪ RetrieveCluster(G, σ, μ″)
                end for
            end if
        end for
    end if
end if
return S
```

6.3. Retrieving Adjacency Relations

Adjacency relation $A_p(\sigma)$ for a p-simplex σ, when $p > 0$ is retrieved by first extracting the $p+1$ $(p-1)$-faces of σ, and then retrieving, for each $(p-1)$-face σ' of σ, co-boundary relation $C_{p-1,p}(\sigma')$. For $p = 0$, adjacency relation $A_0(v)$ for a vertex v is obtained by first retrieving the set of edges in co-boundary relation $C_{0,1}(v)$, and then retrieving the other extreme vertex of each edge e in $C_{0,1}(v)$ through boundary relation $B_{1,0}(e)$. Algorithm **Adjacency** provides a pseudo-code description of the technique for retrieving adjacency relations.

For $p > 0$, the worst-case time complexity of the algorithm for adjacency relation $A_p(\sigma)$ is equal to $O(p\mathcal{C}^p_{p-1})$, and, thus, it is linear in the total number of simplices incident at the p-faces of σ. The time complexity of the algorithm for retrieving $A_0(v)$ is equal to $O(\mathcal{C}^1_0)$, and, thus, it is linear in the number of top simplices incident at vertex v.

7. Comparisons

In this section, we compare the space complexity and the performances of the simplified incidence graph with those of the incidence graph for general d-dimensional simplicial complexes, and of the indexed data structure with adjacency for d-dimensional simplicial pseudo-manifolds. We also compare two- and three-dimensional instances of the SIG with specific data structures for two-dimensional and three-dimensional simplicial complexes, respectively, which cannot be generalized to higher dimensions. The notations used in this section are the same as those defined in Subsection 4.2.

7.1. Comparison with the Incidence Graph

Both the SIG and the Incidence Graph (IG) store boundary relations $B_{p,p-1}(\sigma)$ for all p-simplices σ, $0 < p \leq d$, in the complex. The total cost of encoding such boundary relations is equal to $\sum_{0<p\leq d}(p+1)n_p$ integers.

The IG stores also co-boundary relations $C_{p,p+1}(\sigma)$ for every p-simplex σ, where $0 \leq p < d$. Encoding the indexes of the simplices in all such co-boundary relations requires the same amount of space as encoding all boundary relations, i.e., $\sum_{0<p\leq d}(p+1)n_p$ integers. Moreover, since co-boundary relations are not constant, they are stored in variable-sized arrays, requiring an additional $\sum_{0\leq p<d} n_p$ integers for indexing the start and the end of such arrays.

Instead of co-boundary relations $C_{p,p+1}(\sigma)$, the SIG stores partial co-boundary relations $C^*_{p,g}(\sigma)$ for any p-simplex σ, where $0 \leq p < d$ and $g > p$.

Algorithm 7 : Adjacency(G, σ)
given a simplified incidence graph G, and a p-simplex, it retrieves the set of p-simplices that are adjacent to σ.

```
Adjacency(G, σ)
Input : a Simplified Incidence Graph, G,
    a simplex, σ
Output : a set of simplices, A, that are adjacent to σ

let p := dim(σ), A := ∅
if p > 0 then
    for all σ' ∈ B_{p,p-1}(σ) do
        A := A ∪ Coboundary(G, p, σ')
    end for
else
    S := Coboundary(G, 1, σ)
    for all σ' ∈ S do
        A := A ∪ (B_{1,0}(σ') − {σ})
    end for
end if
return A
```

In the manifold case, partial co-boundary relations $C^*_{p,g}$, for all p-simplices, with $p < d-1$, degenerate into relation $C^*_{p,d}$, which has only one element. Thus,

- the SIG uses only $\sum_{0<p\leq d}(p+1)\,n_p + \sum_{0\leq p<d} n_p + 3n_{d-1}$ integers and $\sum_{0\leq p<d} n_p$ bits;
- the IG uses $2\sum_{0<p\leq d}(p+1)n_p + \sum_{0\leq p<d} n_p$ integers.

For $d = 2$, the SIG uses approximately $21n_0$ integers and $4n_0$ bits, while the IG uses $28n_0$ integers, since from Euler's formula we have that $n_1 \approx 3n_0$ and $n_2 \approx 2n_0$. In the case of three-dimensional manifold complexes, it has been shown experimentally that $n_3 \approx 6n_0$ [3], and thus $n_2 \approx 12n_0$ and $n_1 \approx 7n_0$. Thus, in this case, the SIG requires approximately $110n_0$ integers and $46n_0$ bits, while the IG requires $168n_0$ integers.

In the general case, we conjecture that the number of top q-simplices incident at a p-simplex σ is less or equal to the number of $p+1$ simplices incident at σ, where $p < d$.

7.2. Comparison with the Indexed Data Structure with Adjacencies

The Indexed data structure with Adjacencies (IA) can describe d-dimensional simplicial pseudo-manifolds. It encodes vertices and d-dimensional simplices as well as boundary relation $B_{d,0}$ and adjacency relation A_d for all d-simplices. Since the IA data structure is specific for pseudo-manifolds, adjacency relation A_d is bounded by a constant that depends on dimension d. Given a complex with n_0 vertices and n_d d-simplices, both boundary relation $B_{d,0}$ and adjacency relation require $(d+1)n_d$ integers. Thus, the IA data structure requires $2(d+1)n_d$ integers for topological relation in total.

When restricted to pseudo-manifolds, the partial coboundary relations $C_{p,g}$ in the SIG will be empty for $g \neq d$, since the only top simplices in a pseudo-manifold are the d-simplices. Moreover, each $(d-1)$-simplex is on the boundary of one or two d-simplices. Thus, the SIG requires:

- $\sum_{0<p\leq d}(p+1)n_p$ integers for boundary relations
- $\sum_{0\leq p<d} n_p + \kappa^d_{d-1} + n_{d-1}$ integers (where $\kappa^d_{d-1} = 2n_{d-1}$), and $\sum_{0\leq p<d} n_p$ bits for co-boundary relations

If we consider three-dimensional manifold complexes, the storage cost of the IA data structure is equal to $8n_3$ integers, while the cost of the SIG is equal to $n_0 + 3n_1 + 7n_2 + 4n_3$ integers and $2(n_0+n_1+n_2)+n_3$ bits since κ^3_2 is equal to $2n_2$. The cost of the IA data structure is thus approximately $48n_0$, while that of the SIG is approximately $130n_0$ integers and $46n_0$ bits. Thus, the storage space of the SIG, in the manifold case and for $d = 3$, is about 2.7 times the storage space of the IA data structure.

On the other hand, the IA data structure encodes only vertices and d-simplices, and only the boundary relations and A_d relations can be retrieved in time linear in the entities involved in the relation. Retrieving all other relations requires a number of operations which is linear in the total number of simplices in the complex.

7.3. Comparison with the Triangle-Segment Data Structure

The Triangle-Segment (TS) data structure is a data structure for two-dimensional simplicial complexes embedded in E^3. It encodes the vertices, and the triangles in a complex Σ. The TS data structure encodes also the following relations:

- for each vertex v in Σ: relation $A_0(v)$ for each edge incident in v which is a top simplex (called a *wire edge*), and relation $C^*_{0,2}(v)$, which stores one triangle for each 1-connected component of the restricted star of v.
- for each triangle t in Σ: boundary relation $B_{2,0}(t)$, and partial adjacency relation $A^*_2(t)$. which stores, for each edge e of t, the triangles preceding and succeeding t in counterclockwise order around t

Note that $A_0(v)$ is equivalent to $C^*_{0,1}(v)$ relation in the SIG.

The total storage cost for the TS data structure is equal to $n_0 + 9n_2 + \kappa^1_0 + \kappa^2_0$ integers and $n_0 + n_2$ bits, of which $3n_2$ integers are for relation $B_{2,0}$, $6n_2$ integers for relation A^*_2, and $n_0 + \kappa^1_0 + \kappa^2_0$ integers, where κ^1_0 and κ^2_0 denote the total number of wire edges and the total number of 2-connected components in the restricted star of a vertex v, respectively, for relations $C^*_{0,1}(v)$ and $C^*_{0,2}(v)$.

A SIG for a 2-complex embedded in E^3, uses three integers and $\sum_{0\leq p\leq 2} n_p$ bits for topological entities, and $2n_0 + 4n_1 + 3n_2 + \kappa^1_0 + \kappa^2_0 + \kappa^2_1$ integers and $n_0 + n_1$ bits for topological relations.

For a manifold 2-complex, $\kappa_0^1 = 0$, $\kappa_0^2 = n_0$ and $\kappa_1^2 = 2n_1$. Thus, the SIG uses $3n_0 + 4n_1 + 5n_2$ integers and $n_0 + n_1$ bits, which is approximately equal to $25n_0$ integers. In the manifold case, the TS has a storage cost of approximately $20n_0$ integers. Thus, the SIG uses roughly $\frac{5}{4}$ as much storage space with respect to the TS data structure. On the other hand, edges are not explicitly represented in the TS data structure, and only boundary relations and co-boundary relations $C_{0,g}$ can be extracted in optimal time from it.

7.4. Comparison with the Non-Manifold Indexed Data Structure with Adjacencies

Simplicial 3-complexes embedded in E^3 consist of vertices, edges, triangles and tetrahedra. Non-manifold properties exhibited by 3-complexes include the presence of top simplices of dimensions 1 and 2 (which we call *wire edges* and *dangling faces*), the presence of many 1-connected components at vertices, and the presence of many 2-connected components at edges.

The *Non-manifold Indexed Data Structure with Adjacencies (NMIA)*, described in [4], is specialized for three-dimensional simplicial complexes embedded in E^3. The NMIA data structure encodes the vertices, the top 1-simplices (wire edges), the top 2-simplices (dangling faces), and the tetrahedra of a simplicial 3-complex Σ. Boundary relation $B_{p,0}$ is stored for each top p-simplex, with $p = 1,2,3$. For each top 2-simplex or top 3-simplex σ, the simplices preceding and succeeding σ around each of its edges are encoded. One top-simplex is stored for each connected component of the restricted star $st(v)$ of each vertex v.

To evaluate the storage cost of the NMIA data structure, we denote with n'_1 and n'_2 the number of wire edges and dangling faces, respectively. For each vertex v, we denote with $c_1(v)$ the number of connected components in the restricted star $st(v)$, and with $c_1 = \sum_{v\in\Sigma} c_1(v)$ the total number of connected components summed over all the vertices in the complex. The storage cost of the NMIA data structure is equal to $n_0 + n'_1 + n'_2 + n_3$ bits for entities, and $n_0 + 2n'_1 + 9n'_2 + 16n_3 + c_1$ integers for topological relations. $2n'_1 + 3n'_2 + 4n_3$ integers are required for boundary relations, $n_0 + c_1$ integers are required for storing the connected components at vertices, $6n'_2$ integers are necessary for encoding the neighbors around each edge of a top 2-simplex, and $12n_3$ integers are needed for encoding the neighbors around each edge of a 3-simplex.

Since 3-complexes embedded in E^3 are necessarily pseudo-manifolds, the SIG for such complexes can be specialized so that relation $C^*_{2,3}(t)$ for each triangle t needs only to be stored in a fix-sized array of size 2, and the flag that indicates whether the manifold property holds at an entity does not apply to the triangles and tetrahedra of the complex. The storage cost of the SIG for such a complex is equal to 4 integers and $\sum_{0\leq p\leq 3} n_p$ bits for topological entities and $2n_0 + 4n_1 + 5n_2 + 4n_3 + \sum_{g=1,2,3} \kappa_0^g + \sum_{g=2,3} \kappa_1^g$ integers and $n_0 + n_1$ bits for topological relations.

For manifold complexes, the storage cost of the NMIA is approximately $98n_0$ integers since $c_1 = n_0$ and $n'_1 = n'_2 = 0$. The storage cost of the SIG is $n_0 + 3n_1 + 5n_2 + 4n_3 + \kappa_2^3$ integers and $2(n_0 + n_1) + n_2 + n_3$ bits, which is approximately equal to $130n_0$ integers.

8. Concluding Remarks

We have described a new data structure for d-dimensional simplicial complexes, the Simplified Incidence Graph (SIG), and we have presented efficient algorithms for retrieving incidence and adjacency relations from such a representation as well as for generating it from an incidence graph. The SIG is dimension-independent. It is scalable, since it exhibits an overhead equal to n_{d-1} integers, where n_{d-1} denotes the number of $(d-1)$-simplices in the complex, when applied to manifold simplicial complexes. We have also shown that all the co-boundary and the adjacency relations can be retrieved from a SIG in time linear in the number of top simplices incident in the query simplex.

Our comparisons have shown that the SIG is more compact and efficient than the incidence graph, since it exploits the fact that it is not necessary to encode the complete co-boundary relations when dealing with simplicial complexes. It requires more space than an indexed data structure with adjacencies, but this latter is limited to describe pseudo-manifold complexes, and only boundary and A_d relations can be retrieved from it in optimal time. The SIG is also quite compact and efficient when compared with optimized data structures specific for two- and three-dimensional simplicial complexes, which, however, do not describe all simplices in a complex, and from which not all topological relations can be extracted in optimal time.

Current and future developments of this work involve an in-depth comparison with the incidence graph in the general case of arbitrary simplicial complexes in terms of computational efficiency, and the development of update operations, like edge contraction, on a simplicial complex described through a simplified incidence graph.

9. Acknowledgment

This work has been partially supported by the project MacroGeo (*Algorithmic and Computational Methods for Geometric Object Representation*, funded by the Italian Ministry of Education, University and Research under contract number RBAU01MZJ5.

References

[1] B. G. Baumgart. Winged edge polyhedron representation. Technical Report CS-TR-72-320, Stanford

University, Department of Computer Science, October 1972.

[2] E. Brisson. Representing geometric structures in *d* dimensions: topology and order. In *Proceedings 5th ACM Symposium on Computational Geometry*, pages 218–227. ACM Press, 1989.

[3] P. Cignoni, L. De Floriani, P. Magillo, E. Puppo, and R. Scopigno. Selective refinement queries for volume visualization of unstructured tetrahedral meshes. *IEEE Transactions on Visualization and Computer Graphics*, 10(1):29–45, January-February 2004.

[4] L. De Floriani and A. Hui. A scalable data structure for three-dimensional non-manifold objects. In *Proceedings ACM/Eurographics Symposium on Geometry Processing*, pages 73–83, June 2003.

[5] L. De Floriani, P. Magillo, E. Puppo, and D. Sobrero. A multi-resolution topological representation for non-manifold meshes. *Computer Aided Design*, 36(2):141–159, February 2004.

[6] L. De Floriani, F. Morando, and E. Puppo. Representation of non-manifold objects in arbitrary dimension through decomposition into nearly manifold parts. In *8th ACM Symposium on Solid Modeling and Applications*, pages 103–112. ACM Press, June 2003.

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

[8] D. Dobkin and M. Laszlo. Primitives for the manipulation of three-dimensional subdivisions. *Algorithmica*, 5(4):3–32, 1989.

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

[10] B. Falcidieno and O. Ratto. Two-manifold cell-decomposition of R-sets. In A. Kilgour and L. Kjelldahl, editors, *Proceedings Computer Graphics Forum (EUROGRAPHICS '92)*, volume 11, pages 391–404, September 1992.

[11] A. Gueziec, G. Taubin, F. Lazarus, and W. Horn. Converting sets of polygons to manifold surfaces by cutting and stitching. In *Conference abstracts and applications: SIGGRAPH 98*, Computer Graphics, pages 245–245. ACM Press, 1998.

[12] E. L. Gursoz, Y. Choi, and F. B. Prinz. Vertex-based representation of non-manifold boundaries. In M. J. Wozny, J. U. Turner, and K. Preiss, editors, *Geometric Modeling for Product Engineering*, pages 107–130. Elsevier Science Publishers B.V., North Holland, 1990.

[13] K.Weiler. The radial edge data structure: a topological representation for non-manifold geometric boundary modeling. In H.W. McLaughlin J.L. Encarnacao, M.J. Wozny, editor, *Geometric Modeling for CAD Applications*, pages 3–36. Elsevier Science Publishers B.V. (North–Holland), Amsterdam, 1988.

[14] S.H. Lee and K. Lee. Partial-entity structure: a fast and compact non-manifold boundary representation based on partial topological entities. In *Proceedings Sixth ACM Symposium on Solid Modeling and Applications*, pages 159–170. Ann Arbor, Michigan, June 2001.

[15] P. Lienhardt. Topological models for boundary representation: a comparison with *n*-dimensional generalized maps. *Computer Aided Design*, 23(1):59–82, 1991.

[16] H. Lopes and G. Tavares. Structural operators for modeling 3-manifolds. In *Proceedings Fourth ACM Symposium on Solid Modeling and Applications*, pages 10–18. ACM Press, May 1997.

[17] 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.

[18] J. Rossignac and D. Cardoze. Matchmaker: manifold BReps for non-manifold R-sets. In W. F. Bronsvoort and D. C. Anderson, editors, *Proceedings Fifth Symposium on Solid Modeling and Applications*, pages 31–41. ACM Press, June 9–11 1999.

[19] J.R. Rossignac and M.A. O'Connor. SGC: A dimension-independent model for point-sets with internal structures and incomplete boundaries. In M. J. Wozny, J.U. Turner, and K. Preiss, editors, *Geometric Modeling for Product Engineering*, pages 145–180. Elsevier Science Publishers B.V. (North–Holland), Amsterdam, 1990.

[20] Y. Yamaguchi and F. Kimura. Non-manifold topology based on coupling entities. *IEEE Computer Graphics and Applications*, 15(1):42–50, January 1995.