Obtaining cell complexes associated to four dimensional digital objects

Ana Pacheco[1], Jean-Luc Mari[2] and Pedro Real[1]

[1] Universidad de Sevilla
Dpto. Matematica Aplicada I
E.T.S.I. Informatica, Avda. Reina Mercedes, s/n 41012 Sevilla (Spain)
[2] Aix-Marseille 2 Université
Information and System Science Laboratory (LSIS)
ESIL, Campus de Luminy, case 925, 13288 Marseille cedex 9, France
{ampm,real}@us.es, mari@univmed.fr

Abstract In this paper, we determine a cell complex representation of a 80–adjacent doxel-based 4-dimensional object. The homological information of this polyhedral cell complex can be employed to specify topological features and characteristics of a digital object. This homological information (for example, Euler characteristic, homological classification of cycles, homology generators, relations among them...) of a discrete object can be extracted from some specific boundary operators for each cell of an object (see [3]). The different (up to isometry) polyhedral cells are 400 configurations and their local boundary information can be suitably glued for determining the global boundary of an object and consequently, its corresponding homological information. This fact allows us to implement this technique using a look-up table for the different basic configurations and its corresponding boundary operators.

Keywords digital object; cell complex; weighted complete graph.

1 Introduction

A relevant issue in Digital Image Analysis, Computer Graphics, Pattern Recognition and Computer Vision is to find a topology-based non-redundant representation of a discrete object, useful to extract geometrical and topological properties as well as for visualization, boundary extraction and surface generation tasks. Topological information of digital objects can be obtained using homology. One classical way to specify homological information is making use of a "continuous analogy" $K(X)$. In most of cases, $K(X)$ is embedded in $\mathbb{R}^n$ and it is described in terms of a finite CW-complex (also called *cell complex*). In fact, the homological information allows us to describe the degree of connectivity of the associate cell complex $K(X)$. More concretely, the homological information can be extracted in terms of its n–dimensional holes (a 0–hole is a connected component, a 1–hole is a tunnel and a 2–hole is a cavity in a three dimensional cell complex). In Figure 1, we see a full cell representation of a digital object and its homological information.

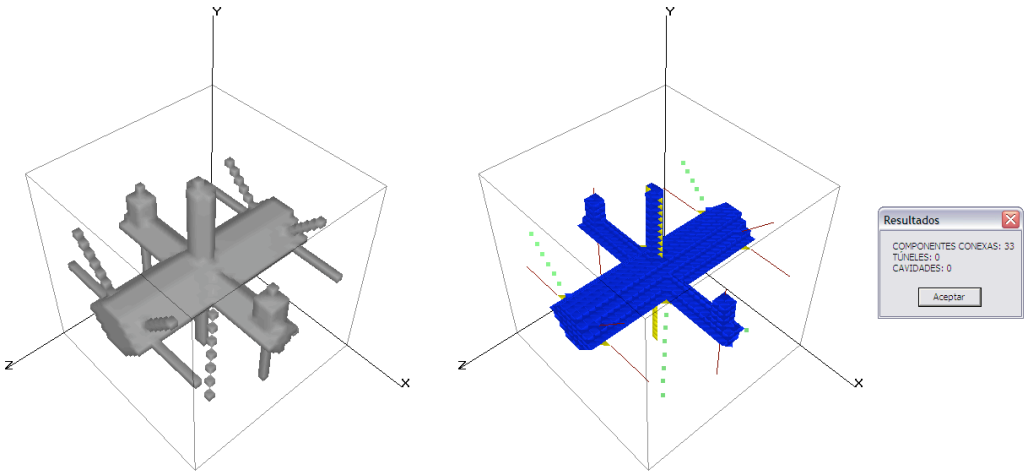

Figure 1: (a) Digital object; (b) Cell complex associated to the digital object, 0–cells, 1–cells, 2–cells and 3–cells are shown in green, brown, yellow and blue respectively; (c) Homological information expressed in terms of connected components (33), tunnels (0) and cavities (0).

Kenmochi et al. present in [11] a procedure to extract topological features from a sequential volume data using a polyhedral object representation and its set operations. Our cell representation approach has as main goal to extract homological information on the 4D discrete object, that can be useful for analysis, recognition and visualization. The stages of the process are the following: (a) to determine a grid divided into hypercubes (cubes of dimension 4) in which the hypervolume is embedded; (b) to assign every point of the grid a binary value (0–points or white points and 1–points or black points); (c) to associate a cell complex to the hypervolume. All the different building blocks (up to isometry) composing the cell complex are specified in this paper in terms of a convex hull and some boundary information of it; and (d) the homology of the associated cell complex can be computed using methods such these: Smith normal form, gradient vector fields, incremental homological algorithm, etc. (see [5, 6, 7] for more details). In this paper, we focus our interest on dealing with the cellularization of the digital object to a fast homological computation. This homological computation is developing in a near future.

2 Preliminaries

In this section is devoted to introduce the main notions that we use along the paper.

A **cell complex** is a set $K = \{K_{(q)}\}_{q\geq 0}$ of cells satisfying two conditions: (1) every face of a cell is a cell; and (2) if σ and σ' are cells, then their intersection is a common face of both. A cell complex can be denoted by a pair (K, ∂) where K is the set of cells which composes the complex and $\partial : \mathbb{Z}[K] \to \mathbb{Z}[K]$ is a linear map called *boundary operator* of K, where $\mathbb{Z}[K]$ is the free abelian group constituted by all the finite linear combination of cells of K. The boundary operator of a p–cell α is a linear combination of all the $(p-1)$–cells that are faces of α, and such that $\partial \circ \partial = 0$. For more details about cell complexes see [9].

A graph can be seen as a cell complex of dimension 1. That is the reason to use graphs in Algorithm 1 to obtain the vertices of the pieces which compose the cell complex associated to the given hypervolume.

A special type of graph is the family of *complete graphs*. A graph is complete if every pair of distinct vertices is connected by a edge. Moreover, if we associate to each edge a weight, we obtain the subfamily of *weighted complete graphs* (used in this paper to determine the vertices of

the pieces which compose the cell complex associated to the given hypervolume). See Figure 2 (a) for an example.

In a natural way, we say G_1 and G_2 are *isomorphic graphs* if exists a bijection between the set of its vertices which preserves adjacencies (if two vertices of G_1 are joined by an edge, then their images by the bijection must be two vertices of G_2 joined by an edge). A stronger concept than isomorphism is isometry. An *isometry* is a distance-preserving map between spaces. For example, the isometries in a 3–dimensional space are rotations, translations and symmetries.

In this paper, we identify a configuration of points on the unit hypercube with its corresponding associated graph constructed in Algorithm 1. In this way, we say two configurations of points on the unit hypercube are isometric if and only if its respective associated graphs are isometric, in the sense of preserving adjacencies and distances.

Information about graph theory can be founded in [8].

An usual tool to represent adjacency relations between vertices of a graph is the *adjacency matrix*. The adjacency matrix of a graph with n vertices is an $n \times n$ matrix $A = (a_{i,j})$ in which the entry $a_{i,j} = m$ if there are m edges from vertex i to vertex j, and it is 0 if there is no edge from vertex i to vertex j. Figure 2 shows an example of a graph together to its adjacency matrix.

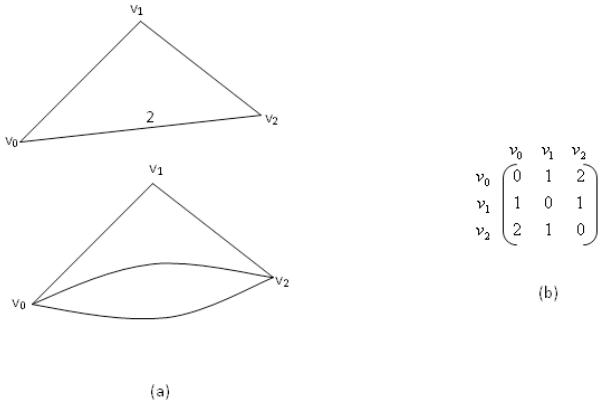

Figure 2: (a) Two representations of the same weighted complete graph (using multiple edges or using weighted edges). (b) The adjacency matrix of the graph.

In order to define adjacency relations, it is necessary to introduce the notion of **neighbor points**.

We say that two points are neighbors if they satisfy a determined distance relation. There are different types of neighborhood; for example, a 3–dimensional point $A = (a_1, a_2, a_3)$ with its 6 neighbors satisfies the relation $N_6 = (a_1 - x)^2 + (a_2 - y)^2 + (a_3 - z)^2 \leq 1$, and with its 26 neighbors satisfies the relation $N_{26} = (a_1 - x)^2 + (a_2 - y)^2 + (a_3 - z)^2 \leq 3$ (see Figure 3).

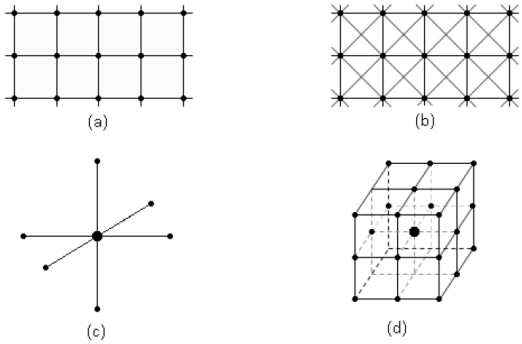

Figure 3: (a) 4–neighborhood in $2D$; (b) 8–neighborhood in $2D$; (c) 6–neighborhood in $3D$; and (d) 26–neighborhood in $3D$.

It is well-known in digital topology (see [12]) that the maximal number of neighbor points of a given n–dimensional point is 3^{n-1}. In consequence, the 8, 26 and 80 adjacency relations are respectively the maximal adjacency for the digital spaces Z^2, Z^3 and Z^4.

3 Grid and neighborhood points

Working with digital objects it is necessary to fix a grid and the relations between the points of this one.

Let us consider Z^4 as the four-dimensional lattice space or discrete space, and the elements of Z^4 the lattice points (see Figure 4).

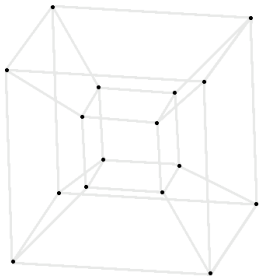

Figure 4: Z^4 space represented by the vertices of a hypercube.

To construct curves, surfaces, volumes and hypervolumes starting from n–simplices in Z^4, we must define a neighborhood between the points which compose the n–simplices. So we can connect the discrete simplices and to construct discrete complexes in Z^4. We can use discrete complexes to represent objects in Z^4. Since discrete complexes can include several parts of different dimensions from 0 to 4, simultaneously, for representing a hypervolume, they can be referred to as multidimensional object representations.

In this paper we work with hypervolumes, so our space of lattice points is Z^4 and our grid is consisting on overlapped $2 \times 2 \times 2 \times 2$ hypercube formed by 16 mutually 80–adjacent (each point has 80 neighbors, it is the maximal adjacency in dimension 4) 4–dimensional points. We can watch an example of this grid in Figure 5.

Then, a thresholding process is applied to the hypervolume embedded in the grid. That hypervolume is subdivided into pieces contained on the hypercubes such way that the vertices

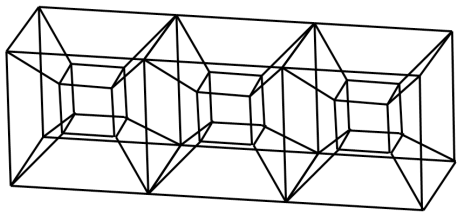

Figure 5: 4–dimensional grid divided into overlapped $2 \times 2 \times 2 \times 2$ 4–dimensional cubes (its intersection is a 3–dimensional cube of 8 mutually 80–adjacent 4–dimensional points) formed by 16 mutually 80–adjacent 4–dimensional points.

of each piece are a finite subset of Z^4. Now, we can obtain the cell complex associated to the hypervolume.

4 Cell complex representation

In this section, we get the cell complex associated to the hypervolume X. Each cell (of any dimension) is contained in an unit hypercube H of the grid. First, we determine the different subsets of vertices of the hypercube and its corresponding convex hulls.

4.1 Determining basic cells

In this subsection, we assume that all the points in Z^4 are assigned binary values (one or zero) and that all points of a finite subset of Z^4 have a value of 1 while all points in its complement of have a value of 0. We recall that points whose values are 1 and 0 are called 1–points or black points and 0–points or white points, respectively.

To obtain the different non-isometric configurations (different finite subsets) of points in the unit hypercube, we develop an algorithm (Algorithm 1) based on graph techniques which consists on associating to each configuration of points a weighted complete graph whose vertices are the vertices of the configuration of points and whose edges are determined by the distance between the points on the hypercube (see Figure 6 for an example).

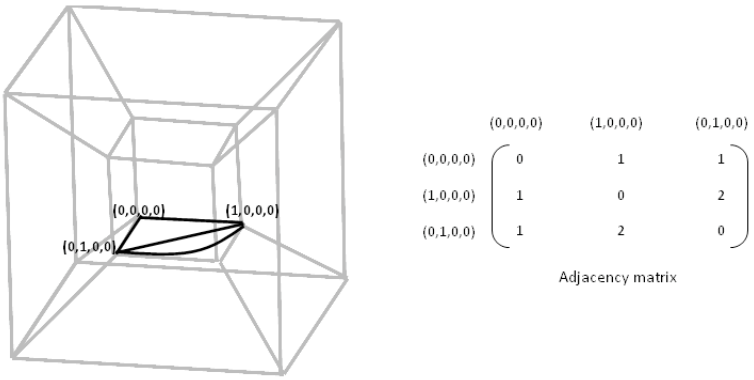

Figure 6: Representation of the graph of vertices $\{(0,0,0,0),(1,0,0,0),(0,1,0,0)\}$ in Algorithm 1.

Algorithm 1

Input: set (C_H) of the 16 vertices of a hypercube.
// Ω: empty list to save vertices of non-isometric graphs.
Output: non-isometric configurations of vertices of an unit hypercube.
begin
  for c=0,...,16 **do**
    Construct an ordered set Ω_c with all the subsets of c vertices of C_H.
    for $\omega \in \Omega_c$ **do**
      G_ω weighted complete graph with adjacency matrix
      $M_\omega = ((m_\omega)_{ij})$ where $(m_\omega)_{ij} = k_{ij}$,
      k_{ij} is the number of different coordinates (distance) between $v_i, v_j \in C_H$
      while $\omega \in \Omega_c$ & $\omega' \in \Omega_c$ & $\omega' < \omega$ **do**
        if G_ω and $G_{\omega'}$ are isometric **then**
          ω and ω' are isometric
          $\Omega_c = \Omega_c - \{\omega\}$
        end if
      end while
    end for
    $\Omega = \Omega \bigcup \Omega_c$
  end for
end

The idea of Algorithm 1 is based on building graphs "embedded" in the unit hypercube. The adjacency matrix of each graph is determined according to the number of different coordinates (distance) between each pair of these vertices.

Algorithm 1 compares the different graphs built starting from configurations of points of the unit hypercube, saving only non-isometric graphs. So, identifying non-isometric configurations of points with their respective associated graphs, we obtain all the non-isometric finite subsets of Z^4 using Algorithm 1 for $c = 0, ..., 16$ (in fact, taking into account complement configurations, it is only necessary to use the algorithm for $c = 0, ..., 8$ or for $c = 8, .., 16$).

The following result is essential to describe our polyhedral cell complex.

Theorem 1. *Two configurations of points on the unit hypercube, C_1 and C_2, are isometric (their corresponding associated graphs are isometric) if and only if its respective convex hulls $CH(C_1)$ and $CH(C_2)$ are isometric.*

Proof. By complements, it is only necessary to prove the result for configurations with c points (for $0 \leq c \leq 8$ or $8 \leq c \leq 16$). We reason for these last ones.

If the convex hulls $CH(C_1)$ and $CH(C_2)$ are isometric, by definition they are isomorphism and the distance between all its points are preserved. Particulary, the distance between the set of its vertices C_1 and C_2 is preserved. Then C_1 and C_2 are isometric.

We suppose $C_1 = (p_i)_i$ and $C_2 = (q_i)_i$ are two isometric configurations with c points for $8 \leq c \leq 16$, and let G_{C_1} and G_{C_2} be their corresponding associated graphs. It exists, by definition, a bijective map $f : (p_i)_i \to (q_i)_i$ with $p_i \in C_1 \subseteq G_{C_1}$ and $q_i \in C_2 \subseteq G_{C_2}$ such that map f preserves the distance between p_i and q_i. We want to prove there is an isometry from R^4 to itself which sends the first configuration to the second one. If $c > 8$, then we can extract a basis starting from the system of vectors $\{p_i - p_0\}_{1 \leq i \leq c}$ (resp. $\{q_i - q_0\}_{1 \leq i \leq c}$), so the result is obtained from linearity. If $c = 8$, then: (a) if we can extract a basis starting from the system of vectors $\{p_i - p_0\}_{1 \leq i \leq 8}$ (resp. $\{q_i - q_0\}_{1 \leq i \leq 8}$), then we reason as before; (b) if we cannot extract a basis starting from the system of vectors $\{p_i - p_0\}_{1 \leq i \leq 8}$ (resp. $\{q_i - q_0\}_{1 \leq i \leq 8}$), then $(p_i)_{1 \leq i \leq 8}$ (resp. $(q_i)_{1 \leq i \leq 8}$) are on the same cube, and the faces of a hypercube are isometric.

□

Once obtained the vertices C of the basic cell $CH(C)$ (the convex hull of these points), the main question is to determine the boundary operator $\partial : Z[CH(C)] \to Z[CH(C)]$ associated to $CH(C)$. With this algorithmic ingredient, we are able to design a look-up table of contractible polyhedral cells which will be the building blocks of the cell complex homologically equivalent to the digital object.

5 Boundary information of the closed cells

In this section we obtain the boundary operator for each constructed basic cell in the previous section. The idea is that for each configuration of points C on the unit hypercube H, we can fix the set of cells $\mathcal{F}$ on H having at least one point that does not belong to C. Moreover, it is possible to organize this set by pairwise different ordered pairs $(\alpha^{(p)}, \beta^{(p+1)})$ $(p = 0, 1, 2, 3)$, such that α is a p–cell that belongs to $\mathcal{F}$ and $\beta^{(p+1)}$ is a $(p+1)$–cell such that α is a $face$ of β. An algorithm to decompose $\mathcal{F}$ in this way can be designed in a simple manner, by taking first the pairs vertices-edges, and then edges-faces, and so on. At the end of the process, we cover all $\mathcal{F}$ (an even number of cells). The next step is to define the combinatorial map $\phi : Cell(C) \to Cell(C)$ by $\phi(\alpha^{(p)}) = \beta^{(p+1)}$ for each pair $(\alpha^{(p)}, \beta^{(p+1)}) \in \mathcal{F}$. The linear map $\pi : \mathbb{Z}[H] \to \mathbb{Z}[H]$ is $\pi = Id - d_{_H} \circ \phi - \phi \circ d_{_H}$, in which $\mathbb{Z}[H]$ is the free abelian group of finite integer linear combination of cells constituting the unit hypercube H; Id is the identity map; and $d_{_H}$ is the boundary operator associated to the unit hypercube cell complex H. $\pi(H) = \{x$ such that $x = \pi(y)$ for some y$\}$ can be seen as a cell complex. Establishing a chain contraction $(H, \pi(H), \pi, g, \phi)$ (where g is the inclusion map) between this cell complex and the unit hypercube, we obtain the boundary operator of $\pi(H)$. Concretely, the boundary operator of $\pi(H)$ is determined by the formula $\partial = \pi \circ d_{_H} \circ g = \pi \circ d_{_H}$. Let us note that the boundary operator $d_{_H}$ can be described using the following result (see [5]):

Lemma 1. *Let $H = L_1 \times L_2 \times L_3 \times L_4$ be a hypercube written as the cartesian product of four segments. The boundary operator of H is given by the formula:*

$$d_{_H} = \partial L_1 \times L_2 \times L_3 \times L_4 - L_1 \times \partial L_2 \times L_3 \times L_4 + L_1 \times L_2 \times \partial L_3 \times L_4 - L_1 \times L_2 \times L_3 \times \partial L_4$$

Summarizing, the couple $(\pi(H), \partial)$ constitutes a basic closed cell and this information can be saved for all the different point configurations on the unit hypercube H. See Figure 7 for an example.

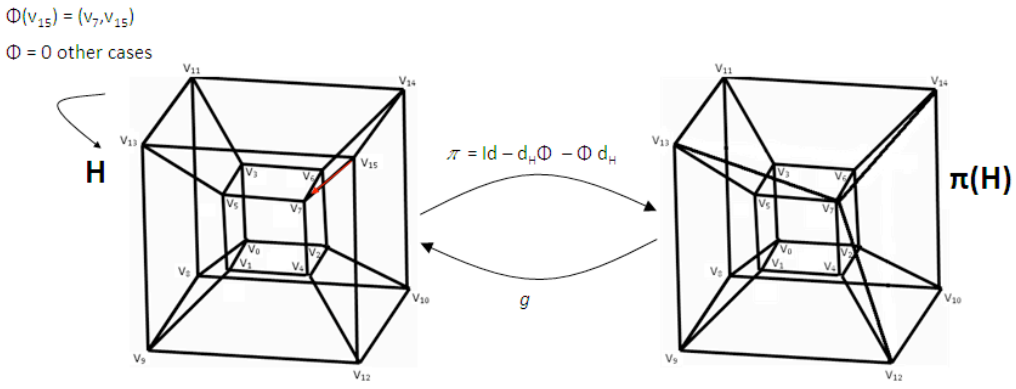

Figure 7: The boundary operator of the basic cell C is given by $\partial = \pi \circ d_{_H} = (v_7, v_5, v_4, v_{13}, v_{12}, v_7, v_{15}, v_1) + (v_{11}, v_{10}, v_6, v_{14}, v_8, v_3, v_2, v_0) + (v_{11}, v_6, v_5, v_{14}, v_{13}, v_7, v_{15}, v_3) - (v_{10}, v_9, v_4, v_{12}, v_8, v_2, v_1, v_0) + (v_{10}, v_6, v_4, v_{14}, v_{12}, v_7, v_{15}, v_2) - (v_{11}, v_9, v_5, v_{13}, v_8, v_3, v_1, v_0) - (v_6, v_5, v_4, v_7, v_3, v_2, v_1, v_0) + (v_{11}, v_{10}, v_9, v_{14}, v_{13}, v_{12}, v_{15}, v_8)$.

6 Conclusion and results

In this paper, we have shown a process of obtaining a cell representation of a 4–dimensional digital object. Once specified the cell complex, homological information (Betti numbers, homology generators and relation between them, cohomology algebra...) can be obtained using methods like those employed in [5, 6, 7].

Four hundred configurations have been obtained when running Algorithm 2 using the symbolic computation package Mathematica. These configurations are the pieces composing a 4–dimensional cell complex associated to a hypervolume. Advanced homological information (including homology generators and relations between them) can be obtained using local boundary information of the basic hyperpolyhedral cells [5, 6, 7].

References

[1] Lorensen W. E. and Cline H. E. Marching cubes: A high-resolution 3d surface construction algorithm. *Computer Graphics*, 21:163–169, 1988.

[2] Molina-Abril H., Pacheco A., and Real P. Homology-based cellularization for digital objects. *Submitted to Pattern Recognition Letters*, 2010.

[3] Molina-Abril H. and Real P. Cell AT-models for digital volumes. *LNCS*, 5534:314–323, 2009.

[4] Mari J-L. and Real P. Simplicialization of digital volumes in 26–adjacency: Application to topological analysis. *Pattern Recognition and Images Analysis*, 19:231–238, 2009.

[5] Kacynski K., Mischaikov, and Mrozek M. *Computational Homology*. Springer-Verlag, 2004.

[6] Mrozek M. and Batko B. Coreduction homology algorithm. *Discrete and Computational Geometry*, 41:96–118, 2009.

[7] Mrozek M., Pilarczyk P., and Zelazna N. Homology algorithm based on acyclic subspace. *Computers and Mathematics with Applications*, 55:2395–2412, 2008.

[8] Diestel R. *Graph Theory*. Springer-Verlag, Heidelberg, 2005.

[9] Fristch R. and Piccinini R. A. *Cellular structure in topology*. Cambridge University Press, 1990.

[10] Gonzalez-Diaz R., Jimenez M. J., Medrano B., Molina-Abril H., and Real P. Integral operators for computing homology generators at any dimension. *LNCS*, 5197:356–363, 2008.

[11] Kenmochi Y., Imiya A., Nomura T., and Kotani K. Extraction of topological features from sequential volume data. *LNCS*, 2059:33–345, 2001.

[12] Kong T. Y. and Rosenfeld A. Digital topology: introduction and survey. *Computer Vision, Graphics, and Image Processing*, 48:357–393, 1989.