Associating Cell Complexes to Four Dimensional Digital Objects

Ana Pacheco and Pedro Real

Dpto. Matematica Aplicada I, E.T.S.I. Informatica,
Universidad de Sevilla,
Avda. Reina Mercedes, s/n 41012 Sevilla (Spain)
http://ma1.eii.us.es/

Abstract. The goal of this paper is to construct an algebraic-topological representation of a 80–adjacent doxel-based 4–dimensional digital object. Such that representation consists on associating a cell complex homologically equivalent to the digital object. To determine the pieces of this cell complex, algorithms based on weighted complete graphs and integral operators are shown. We work with integer coefficients, in order to compute the integer homology of the digital object.

Keywords: digital object, integer homology, integral operator, weighted complete graph.

1 Introduction

Several techniques as magnetic resonance (MR) images and computed tomography (CT) images allow to represent voxel-based digital objects. The homology can be used to obtain topological information of such these objects, but the homological study cannot be made directly over the digital objects. In this sense, it is necessary to apply a thresholding process, in such way, the representation of the voxel-based digital object is made using two colors black and white, which represent (respectively) the elements of the object and the part of the space where such object is not included. The homological study does not only consist on computing Betti numbers, but on determining connected components, "holes", tunnels and cycles (closed curves) in the object. To be extendible previous techniques to higher dimensions is an important goal in computer vision.

Integer homology information of a subdivided $4D$ object (consisting on a set of contractile "bricks" which are glued in a "coherent" manner) is given in this paper by a boundary operator and a "homology" operator for any finite linear combination (with integer coefficients) of bricks, such that, in particular, the boundary of the boundary (resp. the "homology" of the "homology") is zero. Both of them operators can be expressed in terms of arrows acting over the cells of each "brick". For example, the boundary operator (with integer coefficients) of a triangle is an alternate sum of its edges; and the "homology" operator of this triangle is a linear map describing in an algebraic way the contractibility of the triangle to one of its vertices (see Figure 1 for more details).

I. Debled-Rennesson et al. (Eds.): DGCI 2011, LNCS 6607, pp. 104–115, 2011.

Fig. 1. Given a triangle of vertices A, B, C, boundary and "homology" operators can be expressed with arrows defined over its cells: (a) representation in terms of arrows of the boundary operator given by the formula $\partial(ABC) = \partial_0(ABC) - \partial_1(ABC) + \partial_2(ABC) = BC - AC + AB$; and (b) representation in terms of arrows of the "homology" operator (which expresses the contractibility of the triangle to one point) as composition of the integral operators $\phi_1(B) = AB, \phi_2(C) = AC, \phi_3(BC) = ABC$

In this paper, we want to extend the techniques shown in [4–7, 10, 11] to dimension 4. In order to get our goal, we show a method to construct a cell complex homologically equivalent to a 4–dimensional digital object. This cell complex is built piece by piece. The bricks which compose the cell complex are obtained (up to isometry) determining firstly their vertices, and then computing the convex hull of such set of vertices by deforming the unit hypercube with integral operators. In this way, the boundary and contractibility operators of each one of the bricks are inherited of the respective boundary and contractibility operators of the unit hypercube. In Figure 2 we can see the digital object, the associated cell complex and the extracted homological information.

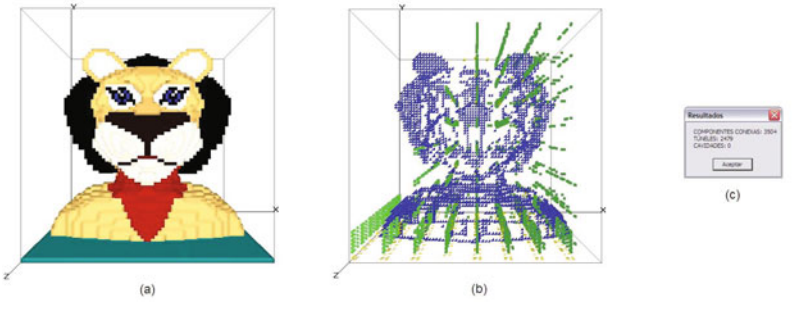

Fig. 2. (a) Digital object; (b) Cell complex associated to the digital object, 0–cells, 1–cells, 2–cells and 3–cells are shown in green, brown, yellow and blue respectively; (c) Homological information expressed in terms of connected components (3504), tunnels (2479) and cavities (0)

The shown method, presents some computational improvements respect to similar methods such as the shown in [8]. Here, unnecessary data corresponding to the simplicialization of the convex hull of the unit hypercube are not to saved, and consequently the convex hull of each one of the bricks is obtained directly. Moreover, the number of integral operator to compute the convex hull of each

brick decreases, since the number of cells of the unit hypercube is smaller than the number of simplices of the simplicialized unit hypercube.

The structure of the paper is the following: in Section 1, we introduce concepts which appear along the paper; in Section 2, we explain our framework and we develop algorithms to associate the cell complex to a given digital object; and finally in Section 3, we show both as several examples of the 402 configurations of points obtained in dimension 4 (applying the algorithms of the previous section) as the way of transferring the boundary and "homology" operators.

2 Preliminaries

As we have commented in Introduction, we show a method to construct (piece by piece) dimensional cell complexes associated to 4–dimensional digital objects using as tools graphs and integral operators. So that, it is necessary to introduce several concepts such these digital objects, cell complexes, isomorphic graphs, integral operators...

The term *digital object* is used for denoting an identifiable item of structured information in digital form within a network-based computer environment. A digital object is a set of sequences of bits or elements, each one of these constitutes structured data interpretable by a computational facility, at least one of the sequences denoting a unique, persistent identifier for that object.

In this paper, we associated to each digital object a mathematical object very used in topology, called cell complex.

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 (or empty). A cell complex is denoted by (K, ∂) where K is the set of cells and ∂ is a map indicating how to join the cells and satisfies $\partial\partial = 0$. The boundary operator shows a relation between cells of different dimensions in order to capture the topology of the cell complex. For example, the boundary of a 1–cell c consists of its two end-points, so using binary coefficients $\partial(c) = A + B$; using integer coefficients, the direction of the edge matters ($AB \neq BA$), so we define $\partial(c) = B - A$. In dimension 2, given a cell complex K whose 0–cells are A, B; 1–cells are $a = AB, b = CB, c = AC$; and the 2–cell is $\tau = ABC$; the boundary is defined as a linear combination of its faces $\partial\tau = AB + BC + CA = a + b - c$.

A graph can be seen as a cell complex of dimension 1, that is, a set of vertices and edges and the relations between them. Graphs are used in Algorithm 1 to obtain the 0–cells of the cell complex associated to a given digital object.

A special type of graphs is the family of *complete graphs*. A graph is complete if every pair of distinct vertices is connected by an edge. Moreover, if we associate to each edge a weight, we obtain the family of *weighted complete graphs* (a graph of this family is associated to each subset of vertices of the unit hypercube in order to determine the non-isometric configurations). See Figure 3 (a) for an example of weighted complete graph.

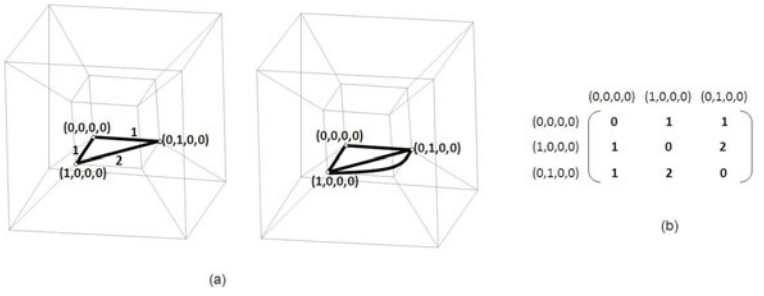

Fig. 3. (a) Two representations of the same situation in terms of graphs (using complete graphs with multiple edges or weighted complete graphs); and its (b) Adjacency matrix

In a natural way, we say G_1 and G_2 are *isomorphic graphs* if there 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).

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 3 shows an example of a graph together to its adjacency matrix.

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.

Information about graph theory can be founded in [1].

Other tools used for obtaining the cell complex associated to a digital object are the integral operators (see [3]).

Given a cell complex (K, ∂) and two cells $a \in K_q$ and $b \in K_{q+1}$ (a is q–face of b), an *integral operator* $\phi_{a,b} : (K_q) \to (K_{q+1})$ is a linear map satisfying: (1) $\phi_{a,b}(a) = b$ and $\phi_{a,b}(c) = 0$ for $c \in K$ different from a; and (2) $\phi_{a,b}\partial\phi_{a,b} = \phi_{a,b}$.

Roughly speaking, an integral operator can be seen as an elementary algebraic thinning operation, allowing the deformation of a cell complex to smaller one (eliminating the cells a and b) with isomorphic homologies. The integral operator $\phi_{a,b}$ can be represented as an arrow from the lower dimension cell a until the higher dimension cell b. In this sense, we say that a cell complex is *contractible* if it has the same topology of a point. For example, the unit hypercube is contractible and its contractibility can be measured in algebraic terms (specifying that the unit hypercube and a point of this one have isomorphic homology groups) by the sequence of integral operators shown in Figure 4.

A *chain contraction* $(f, g, \phi) : (K, \partial) => (K', \partial')$ between two cell complexes is a set of three morphisms $f : \mathcal{C}(K_q) \to \mathcal{C}(K'_q)$ (projection), $g : \mathcal{C}(K'_q) \to \mathcal{C}(K_q)$

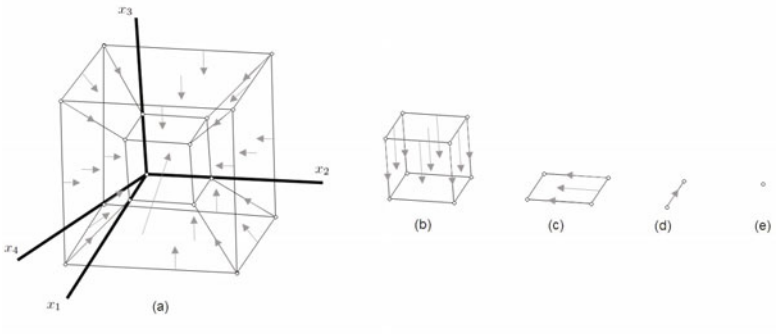

Fig. 4. In order to contract the unit hypercube to the point $(0, 0, 0, 0)$, we define a set of integral operators which are represented by arrows whose: (a) direction coincides with the result of intersecting the hypercube with the hyperplane $x_4 = 0$; (b) direction coincides with the result of intersecting $x_4 = 0$ with $x_4 = x_3 = 0$; (c) direction coincides with the result of intersecting the plane $x_4 = x_3 = 0$ with the line $x_4 = x_3 = x_2 = 0$; and (d) direction coincides with the result of intersecting the line $x_4 = x_3 = x_2 = 0$ with the point $x_4 = x_3 = x_2 = x_1 = 0$. This contraction can be seen as the projection on the coordinate axes

(inclusion) and $\phi : \mathcal{C}(K_q) \to \mathcal{C}(K_{q+1})$ (homotopy operator), where $\mathcal{C}(K_q)$ (resp. $\mathcal{C}(K'_q)$) denotes the set of cell of dimension q of the cell complex (K, ∂) (resp. (K', ∂')) satisfying the following conditions: (a) $\pi = gf = id_C - \partial\phi - \phi\partial$; (b) $fg = id_{C'}$; (c) $f\phi = 0$; (d) $\phi g = 0$; (e) $\phi\phi = 0$.

In Figure 5 we can see an example about integral operators and chain contraction.

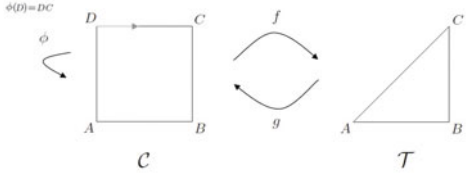

Fig. 5. Integral operators deforming a square $\mathcal{C}$ to a triangle $\mathcal{T}$. The boundary operator of the triangle is computed starting from the boundary operator of the square using the maps which compose the chain contraction as follows $\partial'(\mathcal{T}) = f\partial g(\mathcal{T}) = f\partial(\mathcal{C}) = f\partial(ABCD) = f(AB + BC + CD + DA) = f(AB + BC - DC - AD) = (id - \partial\phi - \phi\partial)(AB+BC-DC-AD) = (AB+BC-DC-AD)-0-\phi(B-A+C-B-C+D-D+A) = AB + BC - DC - AD = AB + BC - (AD + DC)$.

As we have commented in Introduction, the boundary (resp. "homology") operator of the hypercube determines the boundary (resp. "homology") operator of the pieces of the cell complex homologically equivalent to the given digital object. Lemma 1 shows the formula to compute the boundary of a hypercube. See Figure 6 for more details.

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 unit segments. The boundary operator of H is given by* (1)*:*

$$\partial_H = \partial L_1 \times L_2 \times L_3 \times L_4 - L_1 \times \partial L_2 \times L_3 \times L_4 + L_1 \times L_2 \times \partial L_3 \times L_4 - L_1 \times L_2 \times L_3 \times \partial L_4$$

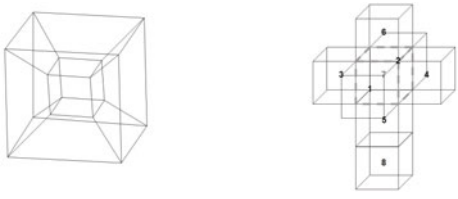

Fig. 6. On left, the tesseract representing a hypercube. On right, the spatial development of a hypercube; the cubes labeled by an odd (resp. even) number have positive (resp. negative) sign in formula (1).

3 Constructing the Cell Complex Associated to a Given 4–Dimensional Digital Object

In this section, the grid and the neighborhood between the points of this one are fixed; and the algorithms to obtain the cell complex associated to a 4–dimensional digital object are developed.

3.1 Establishing the Grid and the Neighboring between the Points

In order to work with digital objects it is necessary to fix a grid as well as the relations between the points of the grid. Our grid is divided into hypercubes (whose intersection is a 3–dimensional cube of 8 mutually 26–adjacent 4–dimensional points) formed by 16 mutually 80–adjacent 4–dimensional points (we work with possible maximal adjacency between points). An example of this division is shown in Figure 7.

This grid is a natural extension to dimension 4 of the grid used in [9], where similar techniques were developed in order to associate cell complexes to 3–dimensional digital objects.

Once established the grid, the initial digital object is embedded in it, so a subdivision into hypercubes of the object is obtained. Applying a thresholding process, we have a digital object subdivided into hypercubes such way that the vertices of each hypercube which are (resp. are not) points of the object are black (resp. white).

Now, the idea is to work each one of these hypercubes with black and white points separately, in order to obtain the pieces of the cell complex associated to the initial object. After, we must join each one of the obtained pieces to compose the cell complex homologically equivalent to the given digital object.

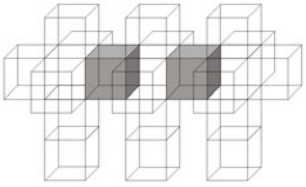

Fig. 7. Spatial development of a sequence of hypercubes formed by 16 mutually 80–adjacent 4–dimensional points, where the intersection between the pairs is a cube of 8 mutually 80–adjacent 4–dimensional points

3.2 Cell Complex Obtention

The main goal of this subsection is the construction of the cell complex associated to a given 4–dimensional digital object. The bricks which compose the cell complex are obtained (up to isometry) determining firstly their vertices (Algorithm 1), and then computing the hull of such set of vertices by deforming the unit hypercube with integral operators (Algorithm 2). In this way, the boundary and "homology" operators of each one of the bricks are inherited of the respective boundary and "homology" operators of the unit hypercube.

In order to determine the vertices of the pieces which compose the complex, we develop an algorithm based on isometric graphs (1–dimensional cell complexes) which allows to compute the non-isometric configurations of c points of the unit hypercube, for $c = 0, ..., 16$. This algorithm associates to each set of points of the unit hypercube a weighted complete graph whose vertices are the points of the set and whose edges are determined by the number of different coordinates between each pair of points. For example, the graph represented in Figure 3 is associated to the set of vertices $\{(0,0,0,0),(1,0,0,0),(0,1,0,0)\}$.

Taking into account previous association is natural to establish Definition 1.

Definition 1. *Two subsets of vertices of the unit hypercube are isometric if and only their respective associated graphs are isometric.*

In order to see the consistency of Definition 1 we must prove Theorem 1.

Theorem 1. *If two subsets of vertices S_1 and S_2 of the unit hypercube are isometric, then there exists a lineal isometry $f : \mathbb{R}^4 \to \mathbb{R}^4$ which sends S_1 into S_2.*

Proof. Taking into account complements (the number of subsets of c vertices of the unit hypercube is the same that the number of subsets of $16 - c$ vertices, with $0 \leq c \leq 16$), it is only necessary to prove the result for subsets with at less 8 vertices of the unit hypercube.

The idea of the proof is the following:

- If $c > 8$, then with the vertices $\{v_0, ..., v_{c-1}\}$ we can construct a basis of $\mathbb{R}^4$ composed by the vectors $\{v_i - v_0\}_{1 \leq i \leq c-1}$, and the problem would be solved by linearity.

- If $c = 8$ and we can construct a basis of $\mathbb{R}^4$ with the vectors $\{v_i - v_0\}_{1 \leq i \leq 7}$, then the problem would be solved by linearity.

- If $c = 8$ and we cannot construct a basis of $\mathbb{R}^4$ with the vectors $\{v_i - v_0\}_{1 \leq i \leq 7}$, then the vertices $\{v_0, ..., v_7\}$ are in a cubic face of the hypercube and all the cubes are isometric.

Using previous ideas, we develop Algorithm 1 which compares the graphs built starting from subsets of points of the unit hypercube, saving only non-isometric graphs. So, identifying non-isometric subsets of points with their respective associated graphs, we obtain all the vertices of the non-isometric bricks which can compose the cell complex associated to a 4–dimensional object. By complement configurations, it is only necessary to use the algorithm for subsets with at less 8 points.

Algorithm 1

Input: set (V_H) of the 16 vertices of the unit hypercube.
// Ω: empty list to save vertices of non-isometric graphs.
Output: non-isometric configurations of vertices of the unit hypercube.
begin
  for c=8,...,16 **do**
    Construct an ordered set Ω_c with all the subsets of c vertices of V_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 between $v_i, v_j \in V_H$
      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
        if G_ω and $G_{\omega'}$ are not isometric **then**
          Update ω'
        end if
      end while
    end for
    $\Omega = \Omega \bigcup \Omega_c$
  end for
end

Note 1. Algorithm 1 is a way to compute marching cube configuration in $4D$.

Once obtained the vertices of the non-isometric bricks which can compose the searched cell complex, the idea is to use integral operators (in a right manner) to deform the unit hypercube in the convex hull of each one of the configurations of points.

Before to show the algorithmic process to define the set of integral operators to deform the unit hypercube, we need to establish a direction over the arrows which represent these integral operators. Indeed, the choice of the direction of

these arrows is arbitrary. We have decided to chose the set of these integral operators as a subset of the sequence (ordered set) of integral operators given in Proposition 1.

Proposition 1. *The sequence of integral operators ϕ_H, which computes the combinatorial contractibility of the unit hypercube to the point $(0,0,0,0)$ is represented by arrows whose directions coincide with the result of:*

- *Intersecting the hypercube with the hyperplane of dimension 3, $x_4 = 0$ (let us note that the result of the intersection is the hyperplane $x_4 = 0$).*
- *Intersecting the hyperplane $x_4 = 0$, with the plane composed by the equations $x_4 = x_3 = 0$.*
- *Intersecting the plane $x_4 = x_3 = 0$, with the line composed by the equations $x_4 = x_3 = x_2 = 0$.*
- *Intersecting the line $x_4 = x_3 = x_2 = 0$ with the point $x_4 = x_3 = x_2 = x_1 = 0$.*

In Figure 4 can be seen a constructive proof of the deformation of the hypercube to the point $(0,0,0,0)$ by ϕ_H.

Once determined the direction of the arrows which represent all integral operators which we can chose to deform each hypercube associated to a configuration of points ω; we show an algorithm which allows us to decide which of these integral operators we must chose to obtain the different pieces which can compose the cell complex associated to the initial object (see Algorithm 2).

Algorithm 2

Input: A configuration of points ω (obtained using Algorithm 1).
Hypercube H_ω associate to a ω configuration.
Sequence of integral operators ϕ_H.
Output: Hull of the points of ω.
begin
  for every 0–cell $\sigma \in H$ **do**
    if $\sigma \notin \omega$ **then**
      $\phi(\sigma) = \phi_H(\sigma)$
    end if
  end for
  for every degenerate cell δ whose 0–cells are in ω **do**
    $\phi(\delta) = \phi_H(\delta)$
  end for
  return $\phi(\omega)$
end
Note: We consider degenerate d–cells as those with at most d $(d-1)$–faces.

Algorithm 2 is divided into two stages: (1) to apply ϕ_H (which computed the contractibility of the unit hypercube to the point $(0,0,0,0)$) to the white points of each hypercube associated to a configuration ω; (2) to eliminate degenerate cells (if they are appeared in previous stage) applying on them ϕ_H.

Note 2. Let us observe that Algorithm 2 determines the hull (non convex hull) of the points. In order to obtain a convex polytope, we must attach an edge for each non-plane face. This process can be made implementing a simple algorithm of recognition of non-plane faces and attaching (in each one of them) an edge whose vertices are the neighbors of the point $p_i \in \omega$ of the non-plane face. Consequently, each non-plane face is transformed into two plane faces.

The pieces which compose the cell complex associated to a given 4–dimensional digital object have been determined (up to isometry). Now, we only need to join these pieces to obtain the searched cell complex. Let us note that such that complex is homologically equivalent to the given 4–dimensional digital object according to Proposition 1 in [3].

4 Conclusions and Results

In this paper, we have shown the process to obtain a cell representation of a 4–dimensional digital object where the topological information is saved. We have obtained 402 configurations (running Algorithm 1 in the symbolic computation package Mathematica) which are the pieces that can compose the cell complex associated to a 4–dimensional digital object. In Figure 8 we can see several examples of the pieces with 14 vertices.

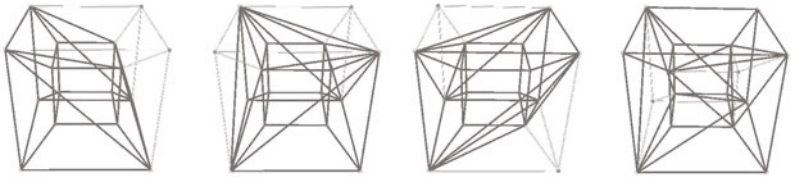

Fig. 8. Convex hulls of the 4 non-isometric bricks of 14 vertices of the unit hypercube

Below, we show with detail each one of the stages of the described process in previous section with one of the 19 non-isometric configurations of 4 vertices of the unit hypercube, and we also show the transference of the boundary (resp. "homology") operator of the unit hypercube to the single one non-isometric configuration of 15 vertices of the unit hypercube.

4.1 Results Obtained for Configurations of 4 Vertices of the Unit Hypercube

Firstly, using Algorithm 1 with $c = 12$ (whose complementary configurations corresponding to the subsets of 4 vertices of the unit hypercube) we obtain 19 non-isometric bricks of 4 vertices.

Now, we must determine the hull of the 19 non-isometric subsets of vertices using Algorithm 2. Particularly, applying Algorithm 2 to the configuration of points $\{\{0,0,0,0\},\{1,0,0,0\},\{0,1,0,0\},\{1,1,1,0\}\}$ (see Figure 9 (a)), we obtain a sequence of integral operators which deforms the unit hypercube in Figure 9 (b).

Let us observe that the obtained cell complex applying Algorithm 2, is not a tetrahedron (convex hull of the subset of points), so we must attach the edge $< (0,0,0,0),(1,1,1,0) >$ as consequence of transforming the non-plane face $< (0,0,0,0),(1,0,0,0),(0,1,0,0),(1,1,1,0) >$ in two plane faces $< (0,0,0,0),(1,0,0,0),(1,1,1,0) >$ and $< (0,0,0,0),(0,1,0,0),(1,1,1,0) >$ (see Figure 9 (c)).

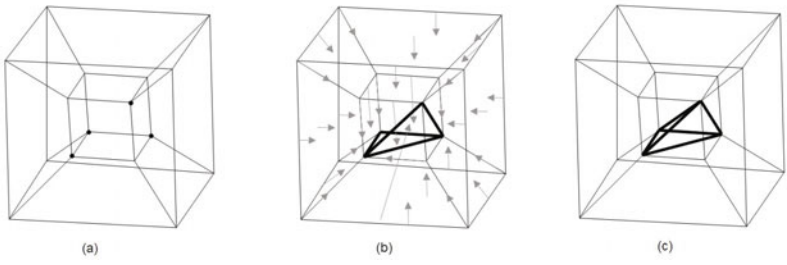

Fig. 9. (a) Vertices of one of the 19 non-isometric configurations of 4 vertices of the unit hypercube; (b) List of integral operators which deform the unit hypercube on the hull of the points; (c) Attaching the edge $< (0,0,0,0),(1,1,1,0) >$ corresponding to the non-plane face $< (0,0,0,0),(1,0,0,0),(0,1,0,0),(1,1,1,0) >$, the convex polytope is obtained.

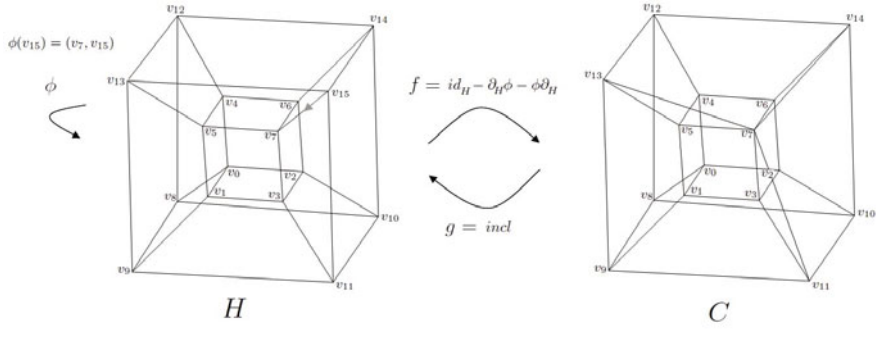

Fig. 10. The boundary operator of C is given by $\overline{\partial}(C) = f\partial g(C) = f\partial(H) = (1-\partial\phi-\phi\partial)(\partial(H)) = -(v_7,v_5,v_4,v_{13},v_{12},v_7,v_{15},v_1)+(v_{11},v_{10},v_6,v_{14},v_8,v_3,v_2,v_0)+(v_{11},v_6,v_5,v_{14},v_{13},v_7,v_{15},v_3) - (v_{10},v_9,v_4,v_{12},v_8,v_2,v_1,v_0) + (v_{10},v_6,v_4,v_{14},v_{12},v_7,v_{15},v_2) - (v_{11},v_9,v_5,v_{13},v_8,v_3,v_1,v_0) - (v_6,v_5,v_4,v_7,v_3,v_2,v_1,v_0) + (v_{11},v_{10},v_9,v_{14},v_{13},v_{12},v_{15},v_8)$, where ϕ operator is obtained using Algorithm 2 and $\partial(H)$ is determined in Lemma 1. In an analogous way, the "homology" operator of C is given by the formula $\overline{\phi}(C) = f\phi_H g(C) = f\phi_H(H) = (1 - \partial\phi - \phi\partial)(\phi_H(H))$, where ϕ operator is obtained using Algorithm 2 and ϕ_H is determined in Proposition 1

4.2 Results Obtained for Configurations of 15 Vertices of the Unit Hypercube

In Figure 10, we show a detailed example where the boundary (resp. "homology") operator of the single non-isometric configuration of 15 points, denoted by C,

is computed starting from the boundary (resp. "homology") operator of the unit hypercube, denoted by H, determined in Lemma 1 (resp. Proposition 1) and the maps of the chain contraction which relates H and C. The boundary (resp. "homology") operator of the other configurations is computed in the same way, using the corresponding maps of the chain contraction which relates the configuration with the unit hypercube.

References

1. Diestel, R.: Graph Theory. Springer, Heidelberg (2005)
2. Fristch, R., Piccinini, R.A.: Cellular structure in topology. Cambridge University Press, Cambridge (1990)
3. Gonzalez-Diaz, R., Jimenez, M.J., Medrano, B., Molina-Abril, H., Real, P.: Integral Operators for Computing Homology Generators at Any Dimension. In: Ruiz-Shulcloper, J., Kropatsch, W.G. (eds.) CIARP 2008. LNCS, vol. 5197, pp. 356–363. Springer, Heidelberg (2008)
4. Herman, G.T.: Discrete multidimensional Jordan surfaces. CVGIP: Graphical Models and Image Processing 54, 507–515 (1992)
5. Kenmochi, Y., Imiya, A., Ichikawa, A.: Boundary Extraction of Discrete Objects. Computer Vision and Image Understanding 71, 281–293 (1998)
6. Kalvin, A.D., Cutting, C.B., Haddad, B., Noz, M.E.: Constructing topologically connected surfaces for the comprehensive analysis of 3D medical structures. In: Proc. of SPIE, vol. 1445, pp. 247–258 (1991)
7. Lorensen, W.E., Cline, H.E.: Marching cubes: A high-resolution 3D surface construction algorithm. Computer Graphics 21, 163–169 (1988)
8. Pacheco, A., Real, P.: Getting topological information for a 80-adjacency doxel-based $4D$ volume through a polytopal cell complex. In: Bayro-Corrochano, E., Eklundh, J.-O. (eds.) CIARP 2009. LNCS, vol. 5856, pp. 279–286. Springer, Heidelberg (2009)
9. Pacheco, A., Real, P.: Polyhedrization, homology and orientation. Progress in Combinatorial Image Analysis, 151–164 (2009)
10. Udupa, J.K.: Multidimensional digital boundaries. CVGIP: Graphical Models and Image Processing 56, 311–323 (1994)
11. Voss, K.: Discrete Images, Objects, and Functions in Z^n. Algorithms and Combinatorics (1987)