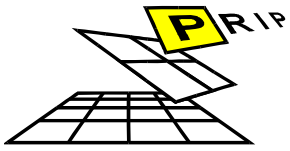
PRIP-TR-111

Computing Homology Group Generators of Images Using Irregular Graph Pyramids

Samuel Peltier, Adrian Ion,
Yll Haxhimusa and Walter G. Kropatsch

Pattern
Recognition &
Image
Processing
Group



Institute of
Computer Aided Automation

Technical Report

Pattern Recognition and Image Processing Group
Institute of Computer Aided Automation
Vienna University of Technology
Favoritenstr. 9/183-2
A-1040 Vienna AUSTRIA
Phone: +43 (1) 58801-18351
Fax: +43 (1) 58801-18392
E-mail: sam@prip.tuwien.ac.at
ion@prip.tuwien.ac.at
yll@prip.tuwien.ac.at
krw@prip.tuwien.ac.at
URL: http://www.prip.tuwien.ac.at/

PRIP-TR-111 December 19, 2006

Computing Homology Group Generators of Images Using Irregular Graph Pyramids[1]

Samuel Peltier, Adrian Ion, Yll Haxhimusa and Walter G. Kropatsch

Abstract

We introduce a method for computing homology groups and their generators of a 2D image, using a hierarchical structure *i.e.* an irregular graph pyramid. Instead of computing homology generators in the base where the number of entities (cells) is large, we first reduce the number of cells by a graph pyramid. Then homology generators are computed efficiently on the top level of the pyramid, since the number of cells is small, and a top down process is then used to deduce homology generators in any level of the pyramid, including the base level *i.e.* the initial image. We show that the new method produces valid homology generators and present some experimental results. In this report we also show that the generators of the first homology groups of a 2D image, computed with this pyramid based method always fit on the borders of the regions.

[1]This work is supported by the Austrian Science Fund under grants P18716-N13 and S9103-N04.

Contents

1 Introduction

Handling 'structured geometric objects' is important for many applications related to geometric modeling, computational geometry, image analysis, etc. One has often to distinguish between different parts of an object, according to properties which are relevant for the application. For image analysis, a region is a (structured) set of pixels or voxels, or more generally a (structured) set of lower-level regions. At the lowest level of abstraction, such an object is a subdivision[1], *i.e.* a partition of the object into cells of dimension 0, 1, 2, 3 ... (called vertices, edges, faces, volumes ...) [1, 2]. In general, combinatorial structures (graphs, combinatorial maps, nG-maps etc.) are used to describe objects subdivided into cells of different dimensions. The structure of the object is related to its decomposition into sub-objects, and to the relations between these sub-objects: basically, topological information is related to the cells and their adjacency or incidence relations. Further information (embedding information) is associated to these sub-objects, and describe for instance their shapes (*e.g.* a point, respectively a curve, a part of a surface, can be associated with each vertex, respectively each edge, each face), their textures or colors, or other information depending on the application. A common problem is to characterize structural (topological) properties of handled objects. Different topological invariants have been proposed, like Euler characteristics, orientability, homology,... (see [3]).

Homology is a powerful topological invariant, which characterizes an object by its '$p-$dimensional holes'. Intuitively the $0-$dimensional holes can be seen as connected components, $1-$dimensional holes can be seen as tunnels and $2-$dimensional holes as cavities. For example, the torus (Figure 1(a)) contains one $0-$dimensional hole, two $1-$dimensional holes (each of them are an edge cycle) and one $2-$dimensional hole (the cavity enclosed by the entire surface of the torus). Even if there are no English term for describing higher dimensional holes, the notion of $p-$dimensional hole is defined in any dimension. Another important property of homology is that local calculations induce global properties. While subdivisions depend on sampling, noise, and data capture, connectivity, cavities and holes are intrinsic properties of objects. Plainly, homology is a tool to study spaces, and has been applied in image processing for 2D and 3D images analysis [4]. Although we use 2D binary image in this report to show the proof of concept, we do not encourage usage of homology groups and generators to find connected components in 2D image, since efficient approaches already exist [5]. However, these 'classical' approaches cannot be easily extended for many problems that exist in

[1]For instance, a Voronoi diagram in the plane defines a subdivision of the plane.

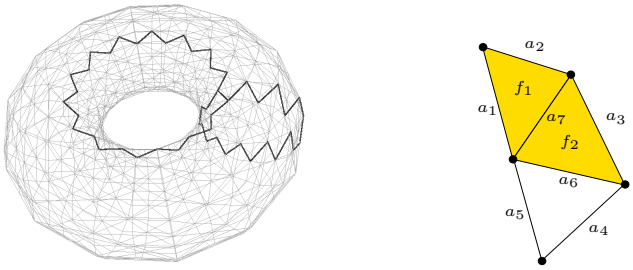


Figure 1: (a) : a triangulation of the torus. (b) : a simplicial complex made of 1 connected component and containing one $1-$dimensional hole.

higher dimensions, since our visual intuition is inappropriate and topological reasoning becomes important. Computational topology has been used in metallurgy [6] to analyse 3D spatial structure of metals in an alloy and in medical image processing [7] in analysing blood vessels. In problems in higher dimensions (*e.g.* beating heart represented in 4D) the importance of homology groups and generators becomes clear in analysing objects (its number of connected components, tunnels, holes, etc) in these spaces, because of the nice formulations which hold in any dimension. One can think of other applications as a preprocessing step to speed up recognition of complex shapes in large image databases, *e.g.* images are first filtered based on their topological invariants and afterward are matched using shapes, appearances, etc.

The usage of homology groups and generators in image processing is a new topic and is not widely spread. In this report we introduce a new method for computing homology groups and their generators using a hierarchical structure which is build by using two operations: contraction and removal. These two operations are used also in [8] to compute incrementally homology groups and their generators of 2D closed surfaces, but a hierarchy is not build. Moreover, if Betti numbers (rank of homology groups) provide the number of 'p-dimensional holes', a set of generators allows to locate them. In [9], it is shown that different parameters influence the geometry of the generators *i.e.* a generator can surround a $p-$dimensional hole more or less closely. We show that the generators built by this new method are always on the borders of regions.

The report is structured as follows. Basic notions on homology and irregular graph pyramids are recalled in Section 2 and 3. The proposed method to compute homology generators is presented in detail in Section 4. We also show that these generators always fit on the borders of the regions. Experimental results on 2D images that show the correctness of the new method are found in Section 5.

2 Homology

In this part, the basic homology notions of chain, cycle, boundary, and homology generator are recalled. Interested readers can find more details in [10, 11, 12].

The homology of a subdivided object[2] X can be defined in an algebraic way by studying the incidence relations of its cells. Within this context, a cell of dimension p is called a $p-$cell and the notion of $p-$*chain* is defined as a sum $\sum_{i=1}^{nb\ p-cells} \alpha_i c_i$, where c_i are $p-$cells of X and α_i are coefficients assigned to each cell in the chain. Homology can be computed using an abelian group $\mathfrak{A}$ for the coefficients α_i. But, the theorem of universal coefficients [10] ensures that all homological information can be obtained by choosing $\mathfrak{A} = \mathbb{Z}$. It is also known [10] that for nD objects embedded in $\mathbb{R}^D$, homology information can be computed considering simply chains with moduli 2 coefficients ($\mathfrak{A} = \mathbb{Z}/2\mathbb{Z}$). Note that in this case, a cell that appears twice on a chain vanishes, because $c + c = 0$ for any cell c when using moduli 2 coefficients. In the following, only chains with coefficients over $\mathbb{Z}/2\mathbb{Z}$ will be considered. Note that the notion of chain is purely formal and the cells that compose a chain do not have to satisfy any property. For example, on the simplicial complex illustrated on Figure 1(b) the sums: a_1+a_4, a_3 and $a_2+a_7+a_4$ are $1-$chains.

For each dimension $p = 0, \ldots, n$, where $n = dim(X)$, the set of $p-$chains forms an abelian group[3] denoted C_p. The $p-$chain groups can be put into a sequence, related by applications ∂_p describing the boundary of $p-$chains as $(p-1)-$chains:

$$C_n \xrightarrow{\partial_n} C_{n-1} \xrightarrow{\partial_{n-1}} \cdots \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0,$$

which satisfy $\partial_p \partial_{p-1}(c) = 0$ for any $p-$chain c. This sequence of groups is called a *free chain complex*.

The boundary of a $p-$chain reduced to a single cell is defined as the sum of its incident $(p-1)-$cells. The boundary of a general $p-$chain is then defined by linearity as the sum of the boundaries of each cell that appears in the chain e.g. in Figure 1(b), $\partial(f_1+f_2) = \partial(f_1)+\partial(f_2) = (a_1+a_2+a_7)+(a_7+a_3+a_6) = a_1 + a_2 + a_3 + a_6$. Note that as mentioned before, chains are considered over $\mathbb{Z}/2\mathbb{Z}$ coefficients and a_7 vanishes as it appears twice in the sum.

For each dimension p, the set of p-chains which have a null boundary are called *p-cycles* and are a subgroup of C_p, denoted Z_p e.g. $a_1 + a_2 + a_7$

[2] In this work we only consider CW-complexes i.e. each cell is homeomorphic to a ball [10].

[3] for any two $p-$chains $c_1, c_2 \in C_p$, $c_1 + c_2 = c_2 + c_1 \in C_p$; for any $c_1, c_2, c_3 \in C_p$, $c_1 + (c_2 + c_3) = (c_1 + c_2) + c_3$; the null chain 0_p is the neutral element; for any $c \in C_p$, there exists $c' \in C_p$ such that $c' + c = c + c' = 0_p$ (in our case, using $\mathbb{Z}/2\mathbb{Z}$ coefficients, for all c we have $c' = c$.

Table 1: Translation of homology notions to graph theory.

Homology theory	Graph theory
0-cell, 1-cell, 2-cell	vertex, edge, face
0-chain, 1-chain, 2-chain	set of vertices, set of edges, set of faces
0-cycle, 1-cycle, 2-cycle	set of vertices, closed path of edges, closed path of faces

and $a_7 + a_5 + a_4 + a_3$ are $1-$cycles. The set of p-chains which bound a $(p+1)-$chain are called *p-boundaries* and they are a subgroup of C_p, denoted B_p e.g. $a_1+a_2+a_7 = \partial(f_1)$ and $a_1+a_6+a_3+a_2 = \partial(f_1+f_2)$ are $1-$boundaries.

According to the definition of a free chain complex, the boundary of a boundary is the null chain. Hence, this implies that any boundary is a cycle. Note that according to the definition of a free chain complex, any $0-$chain has a null boundary, hence every $0-$chain is a cycle.

The p^{th} homology group, denoted H_p, is defined as the quotient group Z_p/B_p. Thus, elements of the homology groups H_p are equivalence classes and two cycles z_1 and z_2 belong to the same equivalence class if their difference is a boundary (*i.e.* $z_1 = z_2+b$, where b is a boundary). Such two cycles are called *homologous* e.g. let $z_1 = a_5+a_4+a_3+a_7$, $z_2 = a_5+a_4+a_6$ and $z_3 = a_1+a_2+a_3$; z_1 and z_2 are homologous ($z_1 = z_2+\partial(f_2)$) but z_1 and z_2 are not homologous to z_3. Let H_p be a homology group generated by q independent equivalence classes $\mathcal{C}_1, \cdots, \mathcal{C}_q$, any set $\{h_1, \cdots, h_q \mid h_1 \in \mathcal{C}_1, \cdots, h_q \in \mathcal{C}_q\}$ is called a *set of generators* for H_p. For example, either $\{z_1\}$ or $\{z_2\}$ can be chosen as a generator of H_1 for the object represented in Figure 1(b).

Note that some of the notions mentioned before could be confused with similar notions from graph theory. Table 1 associates these homology notions with notions classically used in graph theory.

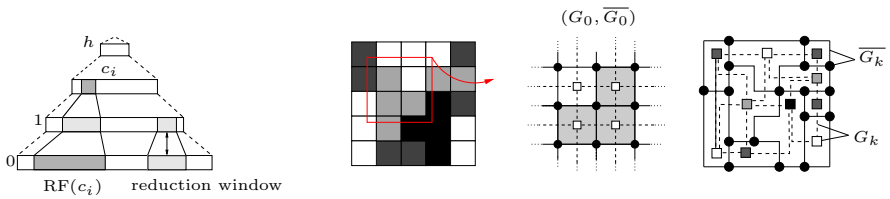
3 Irregular Graph Pyramids

In this part, basic notions of pyramids like receptive field, contraction kernel, and equivalent contraction kernel, are introduced. For more details see [13].

A pyramid (Figure 2(a)) describes the contents of an image at multiple levels of resolution. A high resolution input image is at the base level. Successive levels reduce the size of the data by a *reduction factor* $\lambda > 1.0$. The *reduction window* relates one cell at the reduced level with a set of cells in the level directly below. The contents of a lower resolution cell is computed by means of a *reduction function* the input of which are the descriptions of

the cells in the reduction window. Higher level descriptions should be related to the original input data in the base of the pyramid. This is done by the *receptive field* (RF) of a given pyramidal cell c_i. The RF(c_i) aggregates all cells (pixels) in the base level of which c_i is the ancestor.

Each level represents a partition of the pixel set into cells, *i.e. connected subsets of pixels*. The construction of an irregular pyramid is iteratively local [14]. On the base level (level 0) of an irregular pyramid the cells represent single pixels and the neighborhood of the cells is defined by the 4(8)-connectivity of the pixels. A cell on level $k+1$ (parent) is a union of neighboring cells on level k (children). This union is controlled by so called *contraction kernels* (CK) [15], a spanning forest which relates two successive levels of a pyramid. Every parent computes its values independently of other cells on the same level. Thus local independent (and parallel) processes propagate information up and down and laterally in the pyramid. Neighborhoods on level $k+1$ are derived from neighborhoods on level k. Higher level descriptions are related to the original input by the *equivalent contraction kernels* (ECK). A level of the graph pyramid consists of a pair $(G_k, \overline{G_k})$ of plane graphs G_k and its geometric dual $\overline{G_k}$ (Figure 2(b)). The vertices of G_k represent the cells on level k and the edges of G_k represent the neighborhood relations of the cells, depicted with square vertices and dashed edges in Figure 2(b). The edges of $\overline{G_k}$ represent the borders of the cells on level k, solid lines in Figure 2(b), including so called pseudo edges needed to represent neighborhood relations to a cell completely enclosed by another cell. Finally, the vertices of $\overline{G_k}$ (circles in Figure 2(b)), represent junctions of border segments of $\overline{G_k}$. The sequence $(G_k, \overline{G_k})$, $0 \leq k \leq h$ is called irregular (dual) graph pyramid. For simplicity of the presentation the dual $\overline{G}$ is omitted afterward.



(a) Discrete levels (b) Image to dual graphs

Figure 2: (a) Pyramid concept, and (b) representation of the cells and their neighborhood relations by a dual pair of plane graphs at the level 0 and k of the pyramid.

4 Homology Generators in Graph Pyramids

There exists a general method for computing homology groups. This method is based on the transformation of incidence matrices [10] (which describe the boundary homomorphisms) into their reduced form called *Smith normal form*. Agoston proposes a general algorithm, based on the use of slightly modified Smith normal form, for computing a set of generators of these groups [3]. Even if Agoston's algorithm is defined in any dimension, the main drawback of this method is directly linked to the complexity of the reduction of an incidence matrix into its Smith normal form, which is known to consume a huge amount of time and space. Another well known problem is the possible appearance of huge integers during the reduction of the matrix. A more complete discussion about Smith normal algorithm complexity can be found in [16]. Indeed, Agoston's algorithm cannot directly be used for computing homology generators and different kinds of optimisations have been proposed.

Based on the work of [17] and [18], an optimisation for the computation of homology generators, based on the use of sparse matrices and moduli operations has been proposed [9, 19]. In particular, this method avoids the possible appearance of huge integers. The authors also observed an improvement of time complexity dropping from $O(n^2)$ to $O(n^{5/3})$, where n is the number of cells of the subdivision.

An algorithm for computing the rank of homology groups *i.e.* the Betti numbers have been proposed in [20]. The main idea of this algorithm is to reduce the number of cells of an initial object in order to obtain an homologically equivalent object, made of less cells. In some special cases (orientable objects), Betti numbers can directly be deduced from the resulting object. However, this method cannot directly provide a set of generators. Based on this work, an algorithm for computing a minimal representation of the border of a 3D voxel region, from which homology generators can directly be deduced have been defined in [8].

4.1 Homology Computation using Pyramids (HCP)

The HCP method we propose in this report has the same philosophy as the methods of Kaczynski and Damiand [6, 21]: reducing the number of cells of an object for computing its homology. Moreover, we keep all simplifications that are computed during the reduction process by using a pyramid. In this way, homology generators can be computed in the top level of the pyramid, and can be used to deduce generators of any level of the pyramid. In particular,

we show how generators of the higher level can be directly down-projected on the desired level (using equivalent contraction kernels).

Starting from an initial image, we build an irregular graph pyramid. The method we provide here is valid as long as the algorithm used for the construction of the pyramid preserves homology. In particular, we show here that the decimation by contraction kernels, described in Section 3 [15], preserves homology of a subdivided object. Indeed, homology of the initial image can thus be computed in any level of the pyramid, and in particular in the top level where the object is described with the smallest number of cells.

Moreover, we use the notion of receptive field and equivalent contraction kernel, and show that the generators of homology groups of any level of the pyramid can be deduced from those computed on the higher level. Note that in special cases, the higher level of the pyramid may be reduced to exactly a set of generators of the initial image, as shown in [8].

HCP method can be summarized in the following steps:

1. Starting from a labeled image, a graph pyramid $\{G_0, G_1, \ldots, G_k\}$ is built using contraction kernels of cells with the same label.
2. Homology group generators are computed for G_k, using Agoston's method.
3. Homology generators of any level i can be deduced from those of level $i+1$ using the contraction kernels. In particular, we obtain the homology generators of the initial image.

Note that homology generators of the lowest level can directly be deduced from the highest level using the notion of equivalent contraction kernel (arrow $3'$ in Figure 3). Figure 3 illustrates the general method that we propose for computing homology generators of an image.

4.2 Preserving Homology in HCP

The algorithm described in [20] is based on operations of *interior face reduction* that reduce the number of cells of the subdivision. The main idea is to find a $p-$cell a and a $(p+1)-$cell b, such that a is incident to b. Then a and b are removed and the border of the other $p-$cells that were adjacent to a are modified such that the new border $\partial(b')$ is defined as its initial border added with the border of b. Indeed, if a is incident to exactly two $p-$cells b

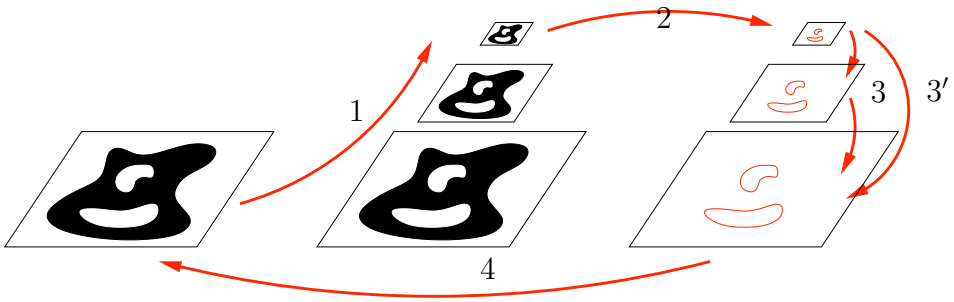


Figure 3: Computing generators of homology groups using an image pyramid (HCP).

and b', the result of the corresponding interior face reduction can be seen as the removal of a and the merging of b and b'. It is proved in [20] that interior face reduction preserves homology.

Observing the dual graph, the operations of contraction and removal that are used to build each level of the pyramid are interior face reductions: two faces that are merged share a common edge that is removed, and an edge is contracted if one of its endpoints is incident to exactly two different edges. Thus, homology is preserved in every level of the pyramid.

4.3 Delineating Generators in HCP

A $1D$ generator in $\overline{G}_k = (\overline{V}_k, \overline{E}_k)$ is a closed path connecting vertices of $\overline{G}_k$ and surrounding at least one hole. Each vertex $\overline{v} \in \overline{G}_k$ is the result of contracting a tree (contraction kernel $\overline{CK}$) of $\overline{G}_{k-1}$. Each edge $(\overline{v}_1, \overline{v}_2) \in \overline{G}_k$ corresponds to a surviving edge $(\overline{w}_1, \overline{w}_2) \in \overline{G}_{k-1}$ with $\overline{w}_1 \in \overline{CK}_{k-1}(\overline{v}_1)$ and $\overline{w}_2 \in \overline{CK}_{k-1}(\overline{v}_2)$ i.e. one that has neither been contracted nor removed[4].

Given a generator in $\overline{G}_k$, mapping it to the level below is done by identifying the surviving edges in $\overline{G}_{k-1}$ corresponding to the generator edges in $\overline{G}_k$ and, where the generator is disconnected, adding paths to fill in the gaps and reconnect. For every two consecutive edges not having a common vertex in $\overline{G}_{k-1}$ but having one in $\overline{G}_k$, the unique path connecting their disconnected endpoints in the contraction kernel $\overline{CK} \subset \overline{G}_{k-1}$ of their shared vertex in $\overline{G}_k$ is added.

Because each path added in $\overline{G}_{k-1}$ is entirely part of a contraction kernel, with contraction being used in the dual only for border simplification purposes, never connecting two different borders, and because the building

[4]Not part of any simplification.

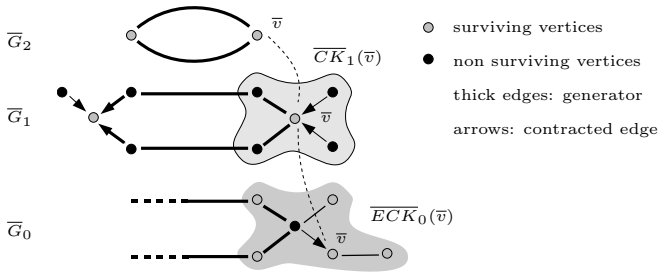


Figure 4: Top-down delineation of a generator computed in $\overline{G}_2$.

process preserves homology (see Section 4.2) the obtained generators will be homologous to the ones in $\overline{G}_k$.

Reiterating this process of mapping the generator cycles of $\overline{G}_k$ from k to $k-1, \ldots$ to 0, cycles in $\overline{G}_0$ corresponding to the generators of the top level can be identified. By replacing the contraction kernels, with the equivalent contraction kernels, using the same methodology, the generator cycles of $\overline{G}_k$ can be directly mapped to $\overline{G}_0$. For an example, see Figure 4.

4.4 Controlling Geometry of the Generators in HCP

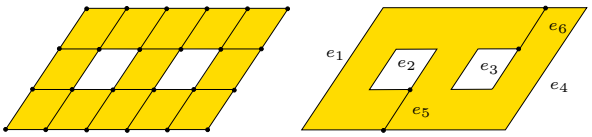
When computing homology generators with Agoston's method, directly on the initial image, we cannot have any control of their geometry. More precisely, the aspect of the obtained generators is directly linked to the construction of incidence matrices, which is determined by the scanning of each cell of the initial image (see [9] for a first study of the influence of different parameters on the geometry of generators).

We prove in this section that for 2D images, the HCP method provides a set of generators that always fit on the borders of a region R. In the following, an edge on the border of a 2D region is called a border edge.

First, we show that any $1-$cycle in the top level of the pyramid computed with HCP method contains only border edges. Second, we show that the down-projection of a $1-$cycle composed of border edges, is still a cycle composed of border edges.

Property 1 *Any $1-$cycle in the top level of the pyramid computed with HCP method contains only border edges*

Proof: On the top level, a region is represented by a unique 2D cell. Hence each edge of the top level is either a border edge or an edge linking two different borders of R (we call it a pseudo edge).



(a) (b)

Figure 5: (a) Bottom level, and (b) top level of the pyramid.

Let z be a $1-$cycle on the top level, if z contains any pseudo edge $e = (v_1, v_2)$, where v_1 and v_2 are two vertices that stand on two different borders of R, then R is made of at least two 2D-cells, which is not possible as any region on the top level is made of only one cell. Hence, any $1-$cycle on the top level of the pyramid contains only border edges.
□

Let us consider Figure 5(b), which represents the top level of the pyramid built from the initial image represented in Figure 5(a). The subdivision is made of one 2D-cell R_1; four border edges e_1, e_2, e_3, e_4; two pseudo edges e_5, e_6; and four vertices. The property 1 ensures that for this subdivision, any $1-$cycle can be written as $\alpha_1 e_1 + \alpha_2 e_2 + \alpha_3 e_3 + \alpha_4 e_4$, where $\alpha_i = 0, 1, i = 1 \ldots 4$.

Property 2 *The delineation of a top level $1-$cycle that lies only on borders results in a $1-$cycle in the bottom level that lies only on borders.*

Proof: A 1D generator can me made out of more than one closed path, in this case its down-projection is done separately for each of them. The process of generator delineation (down-projection) presented in [**?**] requires for each of its closed paths:

- identifying in the bottom level the surviving edges that correspond to the given top level edges
- adding paths connecting two identified edges, if their associated edges from the top level share a vertex (if two identified edges share a common vertex, no path is added).

The identified surviving edges are guaranteed to lie on borders because of their one to one association to their corresponding top level edges.

As presented in Section 4.3, each path added reconnects two consecutive surviving edges, and is a sub-path of the equivalent contraction kernel of the common vertex the two surviving edges share in the top level. Because the equivalent contraction kernels are trees, the added paths are unique [22].

Moreover, these paths lie on borders because:

Table 2: Number of cells on the initial image and on the top of the pyramid.

	Initial image			Top of the pyramid		
	0D-cells	1D-cells	2D-cells	0D-cells	1D-cells	2D-cells
Figure 6.	8153	15785	7630	7	10	1
Figure 7	10352	20148	9793	9	13	1

- in the bottom level, for any two vertices of one border there are exactly two paths that connect them and which are made only of border edges,
- border edges are never removed [**?**] (just contracted or surviving),

we can conclude that the unique path used to reconnect the vertices of two consecutive surviving border edges is made only of border edges. □

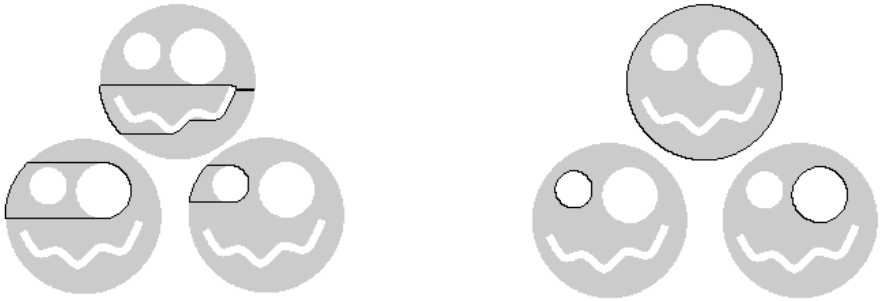
5 Experiments on 2D Images

We present and discuss some experiments that have been performed on 2D images. We compute homology generators, for each region in two different ways: directly on the initial image (bottom level), and on the top level of the pyramid build on this image. Note that for visualisation purposes each edge of a 1D generator is shown by pixels incident to it.

Table 2 shows the number of $0D$, $1D$ and $2D-$cells on the initial image, and on the top level of the pyramid for the shape presented on Figure 6 and Figure 7. One can observe that for each shape the total number of cells is considerably reduced on the higher level of the pyramid. Thus, the computation of homology generators can be done on very smaller matrices on the top level instead of the initial image. In Figure 6 and Figure 7, it can be seen that our new method provides a valid set of generators in each case.

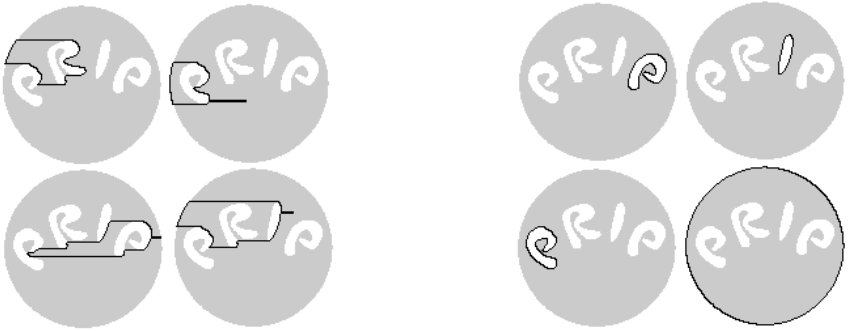
Moreover, using the classical method, we cannot have any control of the geometry of the generators computed. More precisely, the aspect of the obtained generators are directly linked to the construction of incidence matrices, which is determined by the scanning of each cell of the initial image. The shape shown on Figure 8 has been obtained from rotating Figure 6. In Figure 8(a), one can observe that the aspect of the generators computed on the initial image 'follows' the scanning of the cells (from top to bottom, and left to right). The generators obtained in Figure 8(b) always fit on the borders of the objects.

One can note that the set of cycles obtained in Figure 6(a) and Figure 6(b) do not surround the same (set of) $1D-$holes of the shape S. Indeed, these



(a) (b)

Figure 6: Generators overlayed on the image (a): the homology generators computed on the initial image. (b): HCP generators.

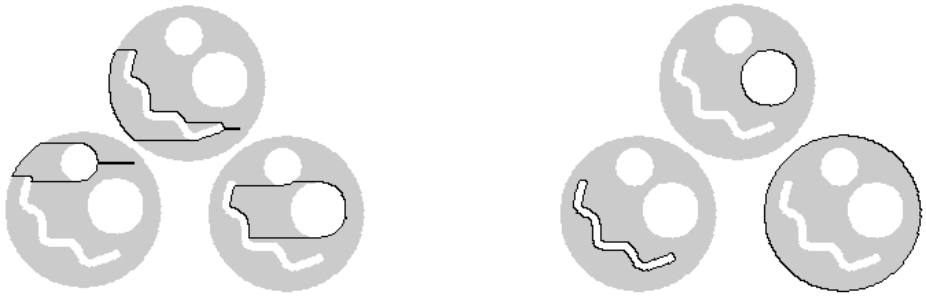


(a) (b)

Figure 7: Generators overlayed on the image (a): the homology generators computed on the initial image. (b): HCP generators.

two sets are two different basis of the same group $H_1(S)$: let a, b and c denote the equivalence class of cycles that surround respectively the left eye, the right eye, and the mouth. The set of generators in Figure 6(a) describe $H_1(S)$ in the basis $\{a+b, c, a\}$ whereas in Figure 6(b), $H_1(S)$ is described in the basis $\{a, a+b+c, b\}$. Note that in this figure we have put one generator (shown in black) per image.

In Figure 9 and Figure 10 some real world images are shown. We have first segmented the images (*e.g.* one can choose minimum spanning tree based pyramid segmentation [23]). In principle one can build generators on these segmented images, but for clarity of this presentation we binary segmented this images (Figure 9(a) and 10(a)). In these binary images the



(a) (b)

Figure 8: Influence of the scanning.

white means 1−dimensional holes. Note that for visualization purpose we show with the gray color an island in Figure 9(a) that is not 1−dimensional hole since it is not enclosed by the black region. The basis in Figure 9(b) and in Figure 9(c) are different but of the same first homology group. The same holds for images Figure 10(b) and Figure 10(c).

The HCP generators shown in Figure 9(c) and Figure 10(b) are nicely fitted on the borders of regions ($1D$−holes). Note that the generators in Figure 9(b), 9(c) and Figure 6(a), 6(b) are shown with white lines, overlaid on the original image.

6 Conclusion

The HCP method for computing homology groups and their generators of images, using irregular graph pyramids has the nice property that the built generators always fit on the borders of the regions in 2D images. Homology generators are computed efficiently on the top level of the pyramid, since the number of cells is small, and a top down process (down-projection) delineates the homology generators of the initial image. Some results have been shown for 2D binary images.

In a future work, we plan to extend this method to 3D and nD images, using the (already existing) structures of 3D and nD irregular pyramids. We also plan to use the property that down-projected generators always fit on borders in order to use homology generators for object matching and object tracking.

References

[1] Kovalevsky, V.A.: Finite topology as applied to image analysis. Computer Vision, Graphics, and Image Processing **46** (1989) 141–161 2

[2] Kovalevsky, V.A.: Digital Geometry Based on the Topology of Abstract Cellular Complexes. In Chassery, J.M., Francon, J., Montanvert, A., Réveillès, J.P., eds.: Géometrie Discrète en Imagery, Fondements et Applications, Strasbourg, France (1993) 259–284 2

[3] Agoston, M.K.: Algebraic Topology, a first course. Pure and applied mathematics. Marcel Dekker Ed. (1976) 2, 7

[4] Allili, M., Mischaikow, K., Tannenbaum, A.: Cubical homology and the topological classification of 2d and 3d imagery. In: Proceedings of International Conference Image Processing. Volume 2. (2001) 173–176 2

[5] Sonka, M., Hlavac, V., Boyle, R.: Image Processing, Analysis and Machine Vision. Brooks/Cole Publishing Company (1999) 2

[6] Kaczynski, T., Mischaikow, K., Mrozek, M.: Computational Homology. Springer (2004) 3, 7

[7] Niethammer, M., Stein, A.N., Kalies, W.D., Pilarczyk, P., Mischaikow, K., Tannenbaum, A.: Analysis of blood vessels topology by cubical homology. In: Proceedings of International Conference Image Processing. Volume 2. (2002) 969–972 3

[8] Damiand, G., Peltier, S., Fuchs, L.: Computing homology for surfaces with generalized maps: Application to 3d images. In: Procedings of 2nd International Symposium on Visual Computing. Volume 4292 of LNCS., Lake Tahoe, Nevada, USA, Springer-Verlag (2006) 1151–1160 3, 7, 8

[9] Peltier, S., Alayrangues, S., Fuchs, L., Lachaud, J.O.: Computation of homology groups and generators. Computers and graphics **30** (2006) 62–69 3, 7, 10

[10] Munkres, J.R.: Elements of algebraic topology. Perseus Books (1984) 4, 7

[11] Zomorodian, A.J.: Topology for computing. Cambridge University Press (2005) 4

[12] Hatcher, A.: Algebraic Topology. Cambridge University Press (2002) available at http://www.math.cornell.edu/∼hatcher/AT/ATpage.html. 4

[13] Jolion, J.M., Rosenfeld, A.: A Pyramid Framework for Early Vision. Kluwer (1994) 5

[14] Meer, P.: Stochastic image pyramids. Computer Vision, Graphics, and Image Processing **45** (1989) 269–294 Also as UM CS TR-1871, June, 1987. 6

[15] Kropatsch, W.G.: Building irregular pyramids by dual graph contraction. IEE-Proc. Vision, Image and Signal Processing **142** (1995) 366–374 6, 8

[16] Kannan, R., Bachem, A.: Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. SIAM Journal on Computing **8** (1979) 499–507 7

[17] Dumas, J.G., Heckenbach, F., Saunders, B.D., Welker, V.: Computing simplicial homology based on efficient smith normal form algorithms. In: Algebra, Geometry, and Software Systems. (2003) 177–206 7

[18] Storjohann, A.: Near optimal algorithms for computing smith normal forms of integer matrices. In Lakshman, Y.N., ed.: Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation, ACM (1996) 267–274 7

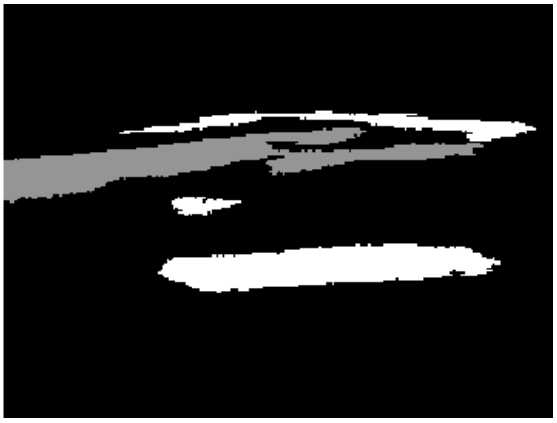
[19] Peltier, S.: Calcul de groupes d'homologie sur des structures simpliciales, simploïdales et cellulaires. Phd thesis, SIC Poitiers, Poitiers, France (2006) 7

[20] Kaczynski, T., Mrozek, M., Slusarek, M.: Homology computation by reduction of chain complexes. Computers & Math. Appl. **34** (1998) 59–70 7, 8, 9

[21] Damiand, G., Peltier, P., Fuchs, L., Lienhardt, P.: Topological map: An efficient tool to compute incrementally topological features on 3d images. In: Proceedings of 11th International Workshop on Combinatorial Image Analysis. Volume 4040. (2006) 1–15 7

[22] Thulasiraman, K., Swamy, M.N.S.: Graphs: Theory and Algorithms. Wiley-Interscience (1992) 11

[23] Haxhimusa, Y., Kropatsch, W.G.: Hierarchy of partitions with dual graph contraction. In Michaelis, B., Krell, G., eds.: Proceedings of German Pattern Recognition Symposium. Volume 2781 of Lecture Notes in Computer Science., Germany, Springer (2003) 338–345 13

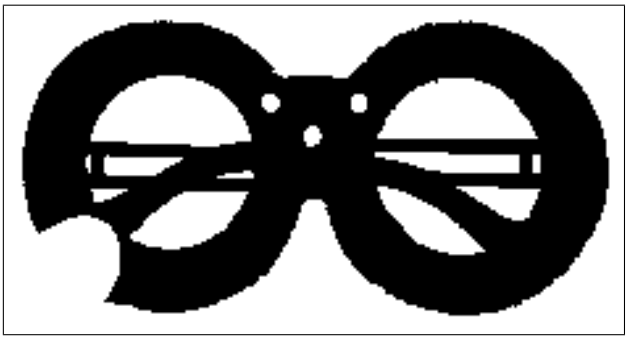


(a)

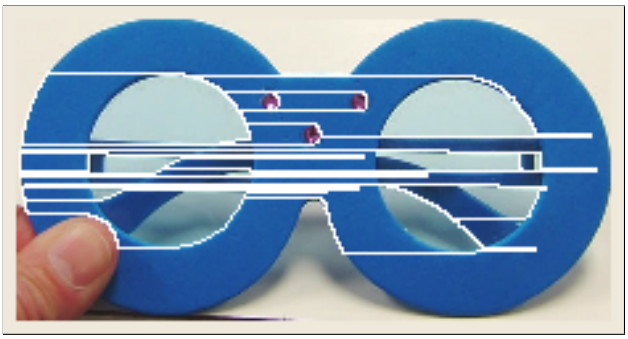
(b)

(c)

Figure 9: (a): segmented image. Generators overlaid on the original image (b): the homology generators computed on the initial image, (c): HCP generators.



(a)



(b)

(c)

Figure 10: (a): segmented image. Generators overlaid on the original image (b): the homology generators computed on the initial image, (c): HCP generators.