Extraction of tiled top-down irregular pyramids from large images.

Romain Goffe, Guillaume Damiand, Luc Brun

▶ **To cite this version:**

Romain Goffe, Guillaume Damiand, Luc Brun. Extraction of tiled top-down irregular pyramids from large images.. 13th International Workshop on Combinatorial Image Analysis (IWCIA'09), Nov 2009, Cancun, Mexico. pp.123-137. hal-00441252v2

HAL Id: hal-00441252

https://hal.science/hal-00441252v2

Submitted on 21 Feb 2011 (v2), last revised 26 Apr 2011 (v3)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Extraction of tiled top-down irregular pyramids from large images

Romain Goffe[1], Guillaume Damiand[2], and Luc Brun[3]

[1] SIC-XLIM, Université de Poitiers, CNRS, UMR6172, Bâtiment SP2MI, F-86962, Futuroscope Chasseneuil, France
goffe@sic.univ-poitiers.fr
[2] LIRIS, Université de Lyon, CNRS, UMR5205, F-69622, Villeurbanne, France
guillaume.damiand@liris.cnrs.fr
[3] GREYC, ENSICAEN, CNRS, UMR6072, 6 Boulevard du Maréchal Juin, F-14050, Caen, France
luc.brun@greyc.ensicaen.fr

Abstract Processing large images is a common issue in the computer vision framework with applications such as satellite or microscopic images. The major problem comes from the size of those images that prevents them from being loaded globally into memory. Moreover, such images contain different information at different levels of resolution. For example, global features, such as the delimitation of a tissue, appear at low resolution whereas finer details, such as cells, can only be distinguished at full resolution. Thus, the objective of this paper is the definition of a suitable hierarchical data structure that would provide full access to all the properties of the image by representing topological information. The idea consists in transposing the notion of tile for top-down topological pyramids to control accurately the amount of memory required by the construction of our model. As a result, this paper defines the topological model of tiled top-down pyramid and proposes a construction scheme that would not depend on the system memory limitations.

Key words: Irregular pyramid; Topological model; Tiled data structure; Combinatorial map;

1 Introduction

The automatic or semi-automatic analysis of high resolution images such as whole slide microscopic images has to face at least two important challenges. Firstly, the size of these images is too important: most of the time, they cannot be stored globally into memory. This drawback induces a decomposition of global images into smaller ones called *tiles*. Secondly, because of the important amount of information present at full resolution, global structures of images only appear at low resolutions. These images naturally induce a mixed multi-resolution and hierarchical representation. An automatic analysis scheme of such images should thus incorporate these constraints.

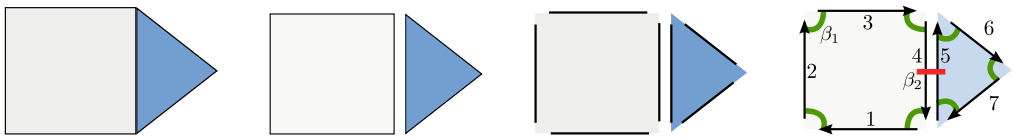
The segmentation phase is the first abstraction step of an image analysis or interpretation algorithm. Usually, the data structures used for encoding partitions

within a segmentation scheme do not provide a full access to all the properties of a partition. These properties include topological information such as the set of neighbors of a region, the set of regions inside a single region or the region containing a given region. The geometrical information includes the geometrical boundary of a region or the geometrical segment between two regions. Topological maps [BMD03, DBF04] have been designed to provide an efficient access to both geometrical and topological information while allowing modifications of the partitions through split and merge operations. However, these models only encode a single partition and thus, do not provide a hierarchy of partitions.

Quadtrees and regular pyramids are the first hierarchical structures used for hierarchical image segmentation. Both structures are based on psycho-visual properties. Quadtrees use the *top-down* notion of focus of attention: segmentation is performed by recursive splitting operations of a given geometrical shape. Regular pyramids use the *bottom-up* concept of reduction window [BCR90]: each pixel of a level corresponds to a larger set of pixels in the level below. However, both structures have several drawbacks [BCR90] concerning their ability to encode connected regions of any size and shape and to provide an efficient access to the neighborhood of a region. The framework introduced by Meer and Montanvert [Mee89, MMR91] allows the definition of bottom-up [JM92, Kro95] irregular pyramids which encode hierarchies of partitions made of connected regions. This pyramidal model provides an efficient access to the neighborhood of any region of the hierarchy. Finally, in order to access both geometrical and topological information, [BK03, SDL06] propose a model of irregular pyramids composed of combinatorial maps.

However, a bottom-up analysis scheme induces at least two problems when processing high resolution images. First, the encoding of the initial partition that corresponds to a very large image may require large amount of memory. This problem is accentuated by the computation of the remaining levels of the pyramid. Second, since these images encode different objects at different resolutions, a better analysis would consist in performing a top-down construction scheme of the pyramid where each region (defined at a given level of resolution) may influence the way its sons (defined at higher resolution) will be processed. Therefore, a top-down construction scheme of irregular pyramids based on topological maps have been defined in [GBD09]. Besides, even when using a top-down analysis scheme, the number of regions defined at a given level of the pyramid is only bounded by the size of the image encoding this level. Since this size may be large for high resolution images, we still have to face the issue of the important amount of memory required by irregular pyramids on high resolution images.

The objective of this paper is the definition of a tiled structure for top-down irregular pyramids. We recall in Sect. 2 the topological models used by our model of tiled top-down pyramid. Section 3 defines the notions of topological tile and tiled pyramid. We detail in Sect. 4 the construction scheme of such a pyramid. Finally, we present some experiments and comparisons in Sect. 5 that highlight involved memory requirements and computational times.



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

Figure 1: Combinatorial maps: construction by successive decompositions. (a) Original image; (b) Decomposed faces; (c) Decomposed edges; (d) 2-Map: arrows represent darts, β_1 and β_2 operators are respectively represented by arcs and segments.

2 Recalls

2.1 Combinatorial Maps

In two dimensions, a combinatorial map (noted 2-map) is a set of vertices, edges and faces that encodes the subdivision and incidence relationships of a topological space [Lie89]. A complete decomposition of an image results in a set of abstract basic elements that we call *darts*. We introduce two operators noted β_i, $i \in \{1, 2\}$ that apply on darts in order to represent adjacency relationships (Fig. 1).

Definition 1 (2-dimensional combinatorial map)**.** *A two-dimensional combinatorial map M (or 2-map) is a triplet $M = (D, \beta_1, \beta_2)$ where:*

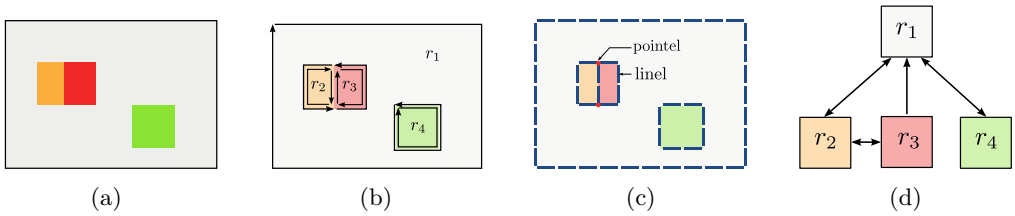
1. *D is a finite set of darts;*
2. *β_1 is a* permutation[1] *on D;*
3. *β_2 is an* involution[2] *on D.*

Intuitively, we can consider a map as a planar graph where β_i operators explicitly define the relationships between edges and where darts allow to differentiate the two extremities of an edge (a dart may be assimilated to a half-edge). In practice, the β_1 permutation allows to turn around a face: it links a dart of the face to the next one encountered while turning in a clockwise orientation. The β_2 involution separates two adjacent regions: it links a dart to the other dart that belongs to the same edge but has an opposite orientation.

2.2 Topological Maps

Topological maps extend the model of combinatorial maps in order to provide a full representation for the partition of an image since a 2-map can only encode the topology of a connected set of objects. The definition of a topological map [BMD03, DBF04] proposes to combine three distinct models: a 2-map that encodes topological relationships, a matrix of *interpixel elements* ([Kov89, KKM90]) that

[1] A *permutation* is a one to one mapping from S onto S.
[2] An *involution* f is a one to one mapping from S onto S such that $f = f^{-1}$.



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

Figure 2: Combinatorial maps: construction by successive decompositions. (a) Original image; (b) Combinatorial map for topology: the image contains four regions $r_i, i \in \{1, 2, 3, 4\}$; (c) Interpixel matrix for geometry: pointels and linels are represented by bold circles and segments and correspond to the embedding of the darts; (d) Tree of regions for inclusion relationships: r_1 contains r_2, r_3 and r_4. r_2 and r_3 are adjacent.

encodes the geometry of the partition elements, and a tree of regions for inclusion relationships.

The combinatorial map encodes the topological relationships thanks to the β_i operators and is *minimal*: the removal of any element would change the topology (Fig. 2(b)).

The geometrical matrix uses pointels, linels and pixels to represent the geometry of a partition. Associating a geometrical information to a topological element is called *embedding*: for example, the embedding of an edge is a sequence of linels (Fig. 2(c)). Pointels, linels and pixels are respectively referred to as i-cells ($i \in \{0, 1, 2\}$) and we call *degree* of an i-cell the number of $(i+1)$-cells touching this cell.

The tree of regions describes inclusion relationships: a region will be the father of the regions it contains (Fig. 2(d)). The root of the tree is called the infinite region. It encodes the background of the image and allows to close the 2-map topologically. The model links darts and regions: a dart knows the region it belongs to and a region knows a representative dart (arbitrary chosen on the external border of the region). Note that the infinite region may be omitted in some figures for visibility reasons.

2.3 Top-down Pyramids

In order to fit the hierarchical framework, the topological map model has been extended to top-down pyramids. Each level G^k of a top-down pyramid is a topological map where each dart and region is connected to its parent in G^{k-1} and its first child in G^{k+1}. These links are called *up/down* relationships. The hierarchy is induced by sets of connected elements having the same parent.

The construction scheme of such a pyramid extracts a first level from a segmentation of the image in few regions (or even a single region). Then, it duplicates the bottom level, links the darts and regions between the two levels and finally, performs splitting operations based on segmentation criteria that transform the level (Fig. 3).

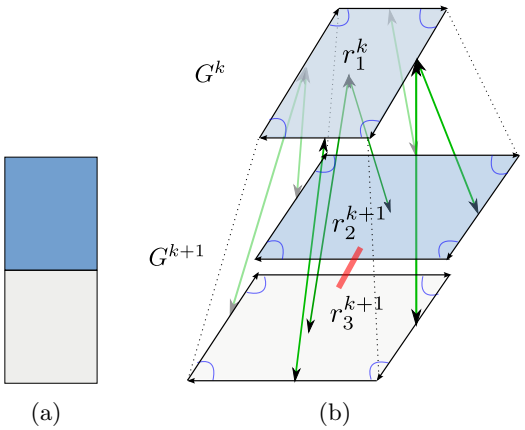


Figure 3: Representation of a top-down pyramid composed of two levels G^k and G^{k+1}. (a) Original image; (b) Pyramid: β_1 and β_2 operators are represented by arcs and segments. Arrows between the levels show up/down relationships between darts and regions. Level G^k is composed of a single region r_1^k. The splitting operation on G^{k+1} allows to differentiate the two regions r_2^{k+1} and r_3^{k+1}.

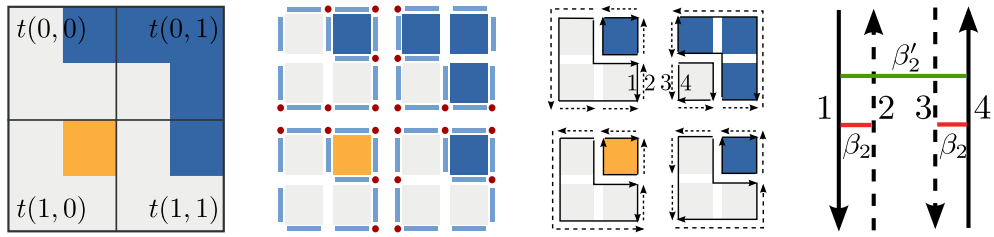
The resolution of the image at level $k+1$ is always greater or equal to the one defined at level k.

If we consider the construction scheme of a top-down pyramid, its main drawback is that for each level, we have to load the whole image associated with this level and represent the whole topological map which encodes the partition defined at this level. Assuming that the image is small enough not to soak memory, the topological map of a level quickly becomes too large as its number of darts and regions increases.

3 Definition of a Tiled Top-down Pyramid Model

In order to define the notion of topological tile, we establish an analogy between the common notion of a tile in image processing: in both cases, a tile represents a regular and arbitrary subdivision of a set and reconstructing the whole initial set is obtained by the juxtaposition of all the tiles it has been split into. Therefore, we define a topological tile as a local topological map composed of a combinatorial map, a geometrical matrix associated with the subdivision and a tree of regions. The topological tile also disposes of additional information specific to the concept of tile.

Since a tile corresponds to an arbitrary subdivision of the image, its border may not mark any real frontier between image components and, in such a case, this border should be considered as fictive. Geometrically, two adjacent tiles share a same edge (Fig. 4(c)). Topologically, we define the β_2' operator on the darts that belongs to the border of a tile so that all the darts of the border of the tile know both their β_2 in the tile (those of the infinite region) and their β_2 in the adjacent tile (Fig. 4(d)).



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

Figure 4: Topological tile. (a) An image divided into 4 tiles; (b) Geometrical representation; (c) Topological representation. β_2' operator is represented by bold segments; (d) Detail of the connection between $t(0,0)$ and $t(0,1)$. Darts 2 and 3 belong to the infinite region. $\beta_2(1) = 2$, $\beta_2(4) = 3$, $\beta_2'(1) = 4$.

Now that the notion of topological tile has been introduced, we can define a hierarchical model based on those tiles, called a tiled top-down topological pyramid. Each level of the pyramid is a set of topological tiles that have the same depth. For each tile, the pyramid assigns a level and a location within the image which is defined at this level. The pyramid also makes the correspondence between the image and the topological maps, handles the fictive borders of the tiles and may swap or load the tiles between memory and disk.

Definition 2 (Tiled top-down topological pyramid)**.** *A tiled top-down pyramid P composed of $l+1$ levels is defined by:*

$P = \{t(i,j,k)\}_{k\in\{0,\dots,l\},(i,j)\in\{0,\dots,W_k\}\times\{0,\dots,H_k\}}$ *where $t(i,j,k)$ is a topological tile and $\forall k,\ 0 \leq k \leq l$:*

(1) $W_k H_k$ and $(i,j) \in \{0,\dots,W_k\} \times \{0,\dots,H_k\}$ encode respectively the number of tiles and the coordinates of one tile at level k.

(2) $t(i,j,k)$ is a topological tile encoding a partition of the geometrical tile (i,j) defined at level k;

(3) $t(i,j,k+1), k < l$ is deduced from $t(i,j,k)$ by performing splitting operations.

Several strategies can be considered for the subdivision in tiles such as a constant size of tiles (in pixels) or a constant number of tiles per level. We will arbitrary choose this last representation since it allows to validate the concept of tiled pyramids and suits the case when the image associated with each level has a constant resolution. The choice of a constant number of tiles per level with a regular subdivision does not impact the irregular pyramid framework of the model: within a tile, the construction scheme remains irregular.

The pyramid defines a hierarchy by introducing up/down relationships between the tiles, similar to the relationships defined on darts and regions by a top-down pyramid. Except for the tiles of top and bottom levels of the pyramid that will not have respectively any parent and any child, each tile knows its single parent (tile

up) and its first child (tile *down*). Note that several tiles of the same level may have the same parent: since all the children of a given tile are adjacent, we can efficiently retrieve all the children of a tile starting from the tile *down* and finding all its neighbors that share the same tile *up*.

The interesting point in dividing the levels of the pyramid in tiles is that most of processings and algorithms require only a few tiles at the same time: we can store apart unused tiles by swapping them on disk so that a maximum of memory space is available for the tiles being processed. A simple solution to record a tile is to apply a unique label for each dart and region that identify them in a file on disk.

We introduce the notion of *local pyramid* which is the set of tiles currently loaded in memory. Since modifications such as splitting or merging operations only apply on the local pyramid, its role is to keep the model coherent by spreading modifications to the other tiles. For example, if we split the border of a tile in memory, the local pyramid updates the adjacent tiles that are either on disk or in memory. Finally, when we load or swap a tile, the local pyramid updates the borders of its neighbors so that adjacent tiles are correctly connected.

4 Construction Scheme of a Tiled Pyramid

4.1 Connection of Adjacent Tiles

If we want a level of the pyramid to be topologically coherent, we have to connect adjacent tiles in order to find darts and regions that belong to several tiles. We will call *basic dart* a dart d whose edge embedding is a single linel. Algorithm 1 describes the operation and can be divided into the three main steps illustrated by Fig. 5.

We will note s (resp. s') the set of darts that are adjacent to t' (resp. t) and that belong to the infinite region of t (resp. t'). First, we split s and s' such as they are only composed of basic darts. This splitting operation ensures that s and s' share the same number of darts (Fig. 5(a)). Second, we link t and t' thanks to the β_2' operator by traversing simultaneously the darts of s and s' (Fig. 5(b)). Finally, since the previous steps may have created degree 2 vertices, we have to perform a simplification pass in order to maintain the minimal property of the model. The idea is to process each vertex that belongs to the shared border (s, s') of two adjacent tiles t and t', and remove it (according to the method in [DL03]) if its embedding is a pointel whose degree equals two in both t and t' (Fig. 5(c)) [3].

4.2 Extraction Process

The global construction scheme of a tiled top-down pyramid starts with the creation of the first level G^0. An arbitrary subdivision of the image defines the geometry of the tiles. Each tile of G^0 is composed of a single region that represents its associated

[3] The vertex_removal operation on a dart d ensures that $\beta_1(\beta_0(d)) \leftarrow \beta_1(d)$ and $\beta_1(\beta_0(\beta_2(\beta_1(d)))) \leftarrow \beta_2(d)$. Then, it deletes d and $\beta_1(\beta_2(d))$.

Algorithm 1: Connecting two adjacent tiles

Data: Two adjacent tiles t and t';
Result: t and t' connected.
Let s the set of darts adjacent to t' and that belong to the infinite region of t;
Let s' the set of darts adjacent to t and that belong to the infinite region of t';
Split s and s' into basic darts;
while $\exists$ *dart* $d \in s | d$ *is unmarked* **do**
    $p \leftarrow pointel(d)$;
    Let d' the dart of s' such as $pointel(\beta_2(d')) = p$;
    $p' \leftarrow pointel(\beta_2(d')$;
    $\beta'_2(\beta_2(d)) \leftarrow \beta_2(d')$;
    if $degree(p) = degree(p') = 2$ **then**
        Vertex_removal(d);
        Vertex_removal(d');
    else
        Mark(d);

subdivision. Then, the creation of a new level only requires a local pyramid composed of four tiles. Indeed, we use an incremental algorithm scanning the subdivisions line by line, starting from the top-left subdivision. A tile $t(i, j, k + 1)$ is defined from $t(i, j, k)$ by performing splitting operations. So, we need in memory the tile *up* $t(i, j, k)$ and the *left* and *top* neighbors $(t(i - 1, j, k + 1), t(i, j - 1, k + 1))$ to connect the left and top borders of $t(i, j, k+1)$. The right and bottom borders of $t(i, j, k+1)$ will be connected when processing respectively $t(i + 1, j, k + 1)$ and $t(i, j + 1, k + 1)$. Note that if $t(i - 1, j, k + 1)$ or $t(i, j - 1, k + 1)$ are not defined, $(i = 0$ or $j = 0)$, the tile $t(i, j, k + 1)$ belongs to the border of the image and does not need to be connected with neighbors. Finally, even in the case when those four tiles are still too large for the memory limitations of the system, we can simply adjust the initial subdivision by reducing the dimensions of the tiles to ensure the feasibility of the process. Algorithm 2 describes the main steps of the extraction of a tiled level.

- `line 1`: Saving a tile records its map, geometry and tree of regions on disk. We have to traverse all darts and regions and save all their connections, geometry and up/down relationships.
- `line 2`: The loading operation of a tile t reads the corresponding file on disk and constructs a new tile before adding it to the pyramid. We first create darts and regions instances in the topological map before filling their properties from reading the corresponding files. Indeed, the operation cannot be performed in a single pass because darts and regions are linked together (a region has a representative dart, a dart knows its region). A region must have been created before it is assigned to a dart and vice versa.

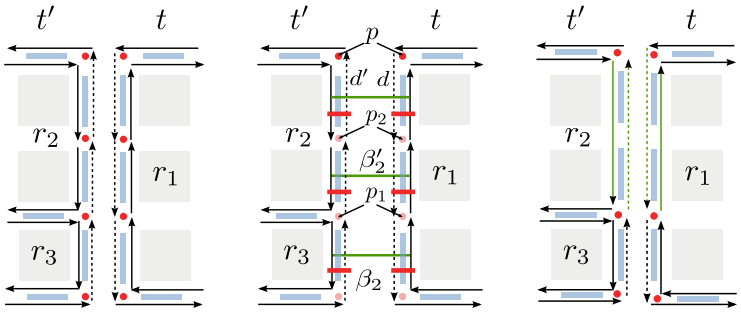


Figure 5: Connection of borders between two tiles. (a) Splitting of the border into basic darts; (b) β'_2 connection: d and d' are two basic darts that belong to r_∞ and $p = pointel(d) = pointel(\beta_2(d'))$. Connecting t and t' comes to link $\beta_2(d)$ with $\beta_2(d')$ such as $\beta'_2(\beta_2(d)) = \beta_2(d')$; (c) Simplification step: $degree(p_1)$ equals 2 in t but 3 in t' whereas $degree(p_2)$ equals 2 in both t and t': the vertex removal operation is only performed on pointel p_2.

- `line 3`: Creating $t = t(i, j, k + 1)$ from $t' = t(i, j, k)$ is similar to adding a level in a top-down pyramid: we first create t as a copy of t', then we set up/down relationships between t and t' and finally, we refine t with a split and merge technique [GBD09]. Note that since the segmentation aspect is not the objective of this paper, we use a criterion based on a basic comparison of average gray levels that was already defined for top-down pyramids. This results in a segmentation which is local to each tile.
- `line 4`: This step connects the left and top neighbors $(t(i-1, j, k), t(i, j-1, k))$ with $t(i, j, k)$ according to the method detailed in Sect. 4.1.
- `line 5`: Since $t(i, j, k)$ has been created, we can record it on disk. It is necessary to save the neighbors too because their borders have been updated in the previous step.
- `line 6`: An unloading operation removes the tile of the local pyramid. Step 5 ensures that the information relative to the tile has been correctly recorded. The operation frees memory so that the size of the local pyramid does not grow too much.

5 Experiments

We have implemented our model of tiled top-down pyramid in C++. Results have been computed on a personal computer with a CPU AMDX2 3800+ (2GHz) and 2GB of RAM on a Linux system. In order to compare results, all the pyramids have the same number of levels (G^k, $k \in \{0, 1, 2, 3\}$) and each level has the same number of tiles (each tile has exactly one parent and one child). All the pyramids

Algorithm 2: Construction of a tiled level

Data: An image I.
Result: A tiled top-down pyramid P composed of $m+1$ levels.
Build first level G^0 of P;
Associate I to P;
foreach *tile* $t \in G^0$ **do**
1 Save t;
for $k = 0$ to $m-1$ **do**
  foreach *tile* $t(i,j,k) \in G^k$ **do**
2 Load $t(i-1,j,k+1)$ and $t(i,j-1,k+1)$;
3 Create $t(i,j,k+1)$ from $t(i,j,k)$;
4 Connect neighbors of $t(i,j,k+1)$;
5 Save $t(i,j,k+1)$, $t(i-1,j,k+1)$, $t(i,j-1,k+1)$ and $t(i,j,k)$;
6 Unload $t(i-1,j,k+1)$, $t(i,j-1,k+1)$ and $t(i,j,k)$;

Table 1: Maximum theoretical memory of the topological model that represents a tile given its size.

tile side (px)	64	128	256	512	1 024	2 048
memory (MB)	1.3	5.2	20.9	83.8	335.5	1 342.1

are composed of four levels which are created with the same splitting method defined in [GBD09]: a basic segmentation criterion merges adjacent regions if the difference of their average grey values is below the thresholds defined as 25/15/5 for levels $G^1/G^2/G^3$. The subdivision of the levels in tiles is regular and tiles and images are squared.

The maximum size of a tile is reached when each pixel corresponds to a different region. In this case, the size of the topology can be estimated by: $4 \times sizeof(dart) \times \sharp pixels + sizeof(region) \times \sharp pixels$. Table 1 presents the required memory for different sizes in our model ($sizeof(dart) = 60\,bytes$ and $sizeof(region) = 80\,bytes$). In practice, the values obtained in our experiments are quite similar because the splitting method bursts regions and produces one single region per pixel before the merging step. Thus, in order to process large images, we choose a fine enough subdivision with tiles smaller than $1\,024 \times 1\,024$. Note that the real amount of memory which is necessary may be more important as we need to store the geometry and the image. Since we can not load the full image in memory, we use the tiled subdivision of the *libtiff* library in order to load only the subdivisions that are necessary to the tiles being processed.

Table 2 gives the computational times and memory usages of top-down pyramids that do not take advantage of tiled subdivision. Indeed, each level is composed of a single tile of the size of the image. The results are computed for successive

Table 2: Extraction of top-down pyramids without tiled subdivision for different scalings of the image *Lena*. Each level is composed of a single tile.

image side (pixels)	extraction time (s)	ram (MB)	darts $G^1/G^2/G^3$	regions $G^1/G^2/G^3$
512	3.7	92.2	70/2 084/47 268	19/460/10 056
1 024	14.8	366.6	74/1 938/47 242	19/428/10 057
2 048	58.2	1 412.3	80/2 010/47 098	20/444/10 027
4 096			na	

Table 3: Extraction of tiled top-down pyramids from large images based on different scalings of the image *Lena*. The subdivision in tiles allows to overcome the memory limitations of the system.

image side (px)	tiles per level	extraction time	ram (MB)	disc (MB)	darts $G^1/G^2/G^3$	regions $G^1/G^2/G^3$
512	1	5 s	92	7	1 212/5 458/54 222	216/989/10 796
2 048	16	53 s	95	20	486/4 332/51 460	77/832/10 494
8 192	256	14 mn	95	272	3 382/9 968/66 764	663/1 757/12 498
32 768	4 096	6 h 38 mn	111	4 315	35 936/48 280/128 858	8 537/10 228/23 309

scalings of the image *Lena*. The second column gives computational times for the whole extraction of the pyramid. The third column is the real memory usage of the application. Fourth and fifth columns give the number of darts and regions for each level. We quickly show the limitations of the top-down model as it cannot process images larger than $2\,048 \times 2\,048$.

Table 3 shows the advantage of using a tiled top-down model compared to the plain top-down extraction. The fourth column represents the size of the whole pyramid on disc (set of files that encode the tiles). We notice that the tiled approach allows to process any large image because we only need to refine the subdivision of the image (smaller tiles) to decrease the amount of required memory.

Table 4 shows the impact of the tiles size during the construction process of a tiled top-down pyramid associated with the image *Lena*. It provides memory

Table 4: Comparison of computational times and memory usages according to the size of the tiles for a tiled top-down pyramid associated to the image *Lena* (1024×1024).

tile side (pixels)	tiles per level	extraction time (s)	ram (MB)	disc (MB)
1 024	1	15.0	366.7	6.8
512	4	14.1	93.4	7.0
256	16	14.2	24.6	7.9
64	256	17.3	2.8	22.0

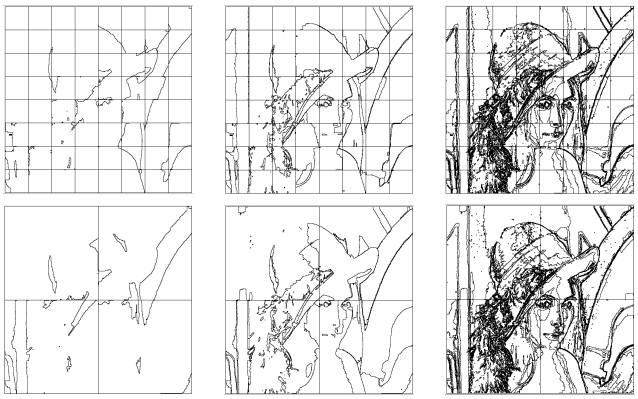


Figure 6: Each row represents the levels G^1, G^2 and G^3 of a tiled top-down pyramid associated with the image *Lena*. First row uses a subdivision of 64 tiles/level, second row has a subdivision of 4 tiles/level.

usage and computational time for four different subdivisions. We can note two interesting points in this experiment. First, the amount of memory required by the local pyramid decreases with smaller tiles. Second, a too fine subdivision increases the computational time of construction and the global amount of memory space. This comes from the additional information required by each tile (labels, up/down relationships, β_2' operators), from the repeated calls of algorithms that connect the tiles and from the increasing number of swapping on disk operations. Therefore, in order to obtain the best extraction time for a given amount of available memory, the size of the tiles should be enlarged as long as the size of the local pyramid does not exceed this amount.

Figure 6 illustrates the tiled construction of a top-down pyramid with a basic segmentation criterion. We can notice that the segmentation may vary although the criterion is the same. Indeed, the split and merge operation that refines a tile is local to this tile and thus, depends on the initial subdivision. Note that fictive borders are represented to illustrate the delimitations of the tiled subdivision. The subdivision has been arbitrary chosen regular (for simplicity and homogeneity between results) but the model is an irregular pyramid since for any tile $t(i, j, k + 1)$ of a tiled top-down pyramid P, the top-down pyramid $P' = \{G^0 = t(i, j, k),\ G^1 = t(i, j, k + 1)\}$ is irregular.

Figure 7 illustrates the tiled construction of a top-down pyramid associated with a multi-resolution image. Indeed, our model easily adapts to multi-resolution images by associating each level of the pyramid to a level of the image. In this example, we use the same number of tiles for each level: we just need to expand the geometrical matrix to make the correspondence between a tile $t(i, j, k)$ and its descendant $t(i, j, k + 1)$.

Figure 7: Extraction of a tiled top-down pyramid from a multi-resolution image based on the image *Cameraman*. First row displays the resolutions of the original image ($16 \times 16, 32 \times 32, 64 \times 64, 128 \times 128, 256 \times 256$). Second row presents the levels G^1 to G^5 of the tiled top-down pyramid (subdivision of 16 tiles/level).

6 Conclusion

In this paper, we have defined a topological model of tiled top-down pyramid. Top-down pyramids are composed of an initial topological map successively refined by splitting operations. The main advantage in extending this model in a tiled structure is to process large images while controlling the memory requirements of the model. Thus, each level of the pyramid is decomposed into a set of topological tiles. A topological tile derives from a topological map and adds complementary information for hierarchical representation between tiles, connecting the border of adjacent neighbors and recording the tile on disk.

We have proposed an algorithm for an incremental construction of a tiled top-down pyramid that ensures that a maximum of four tiles is necessary in the local pyramid (the part of the pyramid that is loaded in memory). Therefore, our experiments have confirmed that we can process any image as long as the size of the local pyramid fits the available memory in the system.

In our future work, we plan to improve the performance of the model with different splitting techniques and multi-threading support. Indeed, the actual model is fast enough for common images but still too slow for very large images although it has shown that the process was viable regarding memory usage. We also intend to compare different strategies of tiled subdivisions in order to improve the association of the pyramid with multi-resolution images. Finally, we will focus on the segmentation aspect in order to define global operations that would not be affected by the local nature of the tiles and would exploit the tiled top-down data structure. This would lead to a structure of tiled top-down irregular pyramid where the irregular aspect is not confined within the tiles.

Acknowledgements

This research is part of the FoGrImMi project, supported by the ANR foundation under grant ANR-06-MDCA-008-01/FOGRIMMI.

References

[BCR90] M. Bister, J. Cornelis, and A. Rosenfeld. A critical view of pyramid segmentation algorithms. *Pattern Recognition Letter.*, 11(9):605–617, September 1990.

[BK03] Luc Brun and Walter Kropatsch. Combinatorial pyramids. *IEEEICIP*, 2:33–37, 2003.

[BMD03] Luc Brun, Myriam Mokhtari, and Jean Philippe Domenger. Incremental modifications on segmented image defined by discrete maps. *Journal of Visual Communication and Image Representation*, 14:251–290, 2003.

[DBF04] G. Damiand, Y. Bertrand, and C. Fiorio. Topological model for two-dimensional image representation: definition and optimal extraction algorithm. *Computer Vision and Image Understanding*, 93(2):111–154, february 2004.

[DL03] G. Damiand and P. Lienhardt. Removal and contraction for n-dimensional generalized maps. In *DGCI*, pages 408–419, Naples, Italy, november 2003.

[GBD09] Romain Goffe, Luc Brun, and Guillaume Damiand. A top-down construction scheme for irregular pyramids. In AlpeshKumar Ranchordas and Helder Araujo, editors, *Fourth International Conference On Computer Vision Theory And Applications (VISAPP'09)*, volume 1, pages 163–170, February 2009.

[JM92] Jean-Michel Jolion and Annick Montanvert. The adaptative pyramid: A framework for 2d image analysis. *CVGIP*, 55(3):339–348, May 1992.

[KKM90] E. Khalimsky, R. Kopperman, and P.R. Meyer. Boundaries in digital planes. *Journal of Applied Mathematics and Stochastic Analysis*, 3(1):27–55, 1990.

[Kov89] V.A. Kovalevsky. Finite topology as applied to image analysis. 46:141–161, 1989.

[Kro95] Walter Kropatsch. Building irregular pyramids by dual graph contraction. *IEE Proceedings - Vision, Image, and Signal Processing*, 142:366–374, December 1995.

[Lie89] P. Lienhardt. Subdivision of n-dimensional spaces and n-dimensional generalized maps. In *Symposium on Computational Geometry*, pages 228–236, Saarbrücken, Germany, 1989.

[Mee89] P. Meer. Stochastic image pyramids. *CVGIP*, 45:269–294, 1989.

[MMR91] Annick Montanvert, Peter Meer, and Azriel Rosenfeld. Hierarchical image analysis using irregular tessellations. *IEEETPAMI*, 13(4):307–316, April 1991.

[SDL06] C. Simon, G. Damiand, and P. Lienhardt. nd generalized map pyramids: definition, representations and basic operations. *Pattern Recognition*, 39(4):527–538, April 2006.