Efficient encoding of n-D combinatorial pyramids

Sébastien Fourey and Luc Brun
GREYC, CNRS UMR 6072, ENSICAEN & University of Caen,
6 bd maréchal Juin F-14050 Caen, France
{Sebastien.Fourey,Luc.Brun}@greyc.ensicaen.fr

Abstract

Combinatorial maps define a general framework which allows to encode any subdivision of an n-D orientable quasi-manifold with or without boundaries. Combinatorial pyramids are defined as stacks of successively reduced combinatorial maps. Such pyramids provide a rich framework which allows to encode fine properties of objects (either shapes or partitions). Combinatorial pyramids have first been defined in 2D, then extended using n-D generalized combinatorial maps. We motivate and present here an implicit and efficient way to encode pyramids of n-D combinatorial maps.

1. Introduction

Pyramids of combinatorial maps have first been defined in 2D [1], and than extended to nD using generalized maps [5]. Such pyramids encode successive subdivisions of orientable but also, in the case of generalized maps, non-orientable quasi-manifolds [6]. Considering the practical case of orientable-only spaces, the authors have defined pyramids of n-maps [4, 2]. Although not the only one, a main motivation for this choice was to address the space complexity issue raised by such pyramids, as a generalized map is roughly speaking twice as much space consuming as a map for a given orientable-space partition.

Since in applications, such as a region-merging, the whole history of a region may be of interest for choices to be made futher on; it must be possible to retrieve efficiently this information from the structure encoding the corresponding pyramid of maps. For that purpose, a first solution is an explicit encoding of each level of the pyramid, which proves to be untractable even for base maps with reasonable size (Subsection 4.2). Another solution, initially proposed in the 2D case [1], is to use the direct link between elements of two consecutives levels of a pyramid (notion of *reduction window*) and its transtitive closure (notion of *receptive field*) to retrieve that information. In fact, we show in this paper that these two notions allow an implicit encoding, or *folding*, of a whole pyramid as the single base map and two arrays of integers, whose sizes are immediately related with the one of the base map.

In a first section, we recall the definition of n-D combinatorial map and the one of the simultaneous cell removal operation within such maps that allows the construction of a pyramid of combinatorial maps. We then present in Section 3 the notions of connecting walks and connecting dart sequences that generalize the notion of reduction window and receptive fields. Eventually, we present in Section 4 the pyramid folding property and the its related space complexity as compared with the explicit encoding.

2. Pyramids of n-D combinatorial maps

An n-*map* ($n \geq 1$) is defined ([6]) as an $(n+1)$-tuple $M = (\mathcal{D}, \gamma_0, \dots, \gamma_{n-1})$ where $\mathcal{D}$ is a finite non-empty set of *darts*, $\gamma_0, \dots, \gamma_{n-2}$ are involutions on $\mathcal{D}$, and γ_{n-1} is a permutation on $\mathcal{D}$; such that $\gamma_i\gamma_j$ is an involution for any $i, j \in \{0, \dots, n-1\}$ with $|i-j| \geq 2$.

Each involution γ_i ($i < n$) sews together i-cells of a subdivision by mapping some darts of each cell one to the other. Thus, in the 3-map depicted in Figure 1, $(1, 2)$ is a cycle of γ_0 as well as $(9, 16)$. These two cycles link the two upper-front vertices of the cube to form an edge. On the orther hand, the cycles $(2, 3)$, $(16, 15)$, and $(17, 24)$ of γ_1 illustrate the fact that γ_1 sews edges.

Any dart d in an n-map belongs to a cell of each dimension i of $\{0, \dots, n\}$. Such an i-cell is defined as the orbit $< \gamma_0, \dots, \hat{\gamma}_i, \dots, \gamma_{n-1} > (d)$ if $i < n$ and $< \gamma_0\gamma_1, \dots, \gamma_0\gamma_{n-1} > (d)$ for $i = n$ (where $< \gamma_0, \dots, \hat{\gamma}_i, \dots, \gamma_{n-1} > (d)$ is the set of images of d by any element of the permutation group generated by the permutations $\gamma_0, \dots, \gamma_{n-1}$ without γ_i. Thus, the upper-front edge (or 1-cell) in Figure 1 is defined as $<\gamma_0, \gamma_2>$ (1), that is $\{1, 2, 9, 16\}$.

In a combinatorial pyramid, each map of the pyramid is defined from the map underneath by the simultaneous removal of cells of dimension at most $n-1$ that are parts of the n-cells boundaries. Thus, regions of a subdivision may be merged by removing $(n-1)$-cells and subsequent removals allow the simplification of a map while preserving the features of interest. The cell removal operation is defined as soon as cells to be removed satisfy several constraints regarding their *local degree*, disjointness, and *regularity* [4]. For short, we simply recall the main condition known since [5], which guarantees that the removal of cells yields a proper map: only i-cells that appear to be *locally* incident to two $(i+1)$-cells should be removed.

Given the set K of all darts of the cells to be removed, the n-map M' obtained by removal of these cells is defined as $M' = (\mathcal{D}' = \mathcal{D} \setminus K, \gamma'_0, \dots, \gamma'_{n-1})$, where the permutations γ''s are defined as follows for any $d \in \mathcal{D}'$ (using our notation $d\gamma_i \overset{def}{=} \gamma_i(d)$) :

- If $i \leq n-2$, $d\gamma'_i = d(\gamma_i\gamma_{i+1}^{-1})^k\gamma_i$
- For $i = n-1$, $d\gamma'_{n-1} = d\gamma_{n-1}^{k+1}$

where k is the smallest integer such that $d\gamma'_i \in \mathcal{D}'$.

By an abuse of notation, we may also denote by K the set of cells to be removed that we call a removal kernel in M. A pyramid of maps with height h is then nothing but a *sequence* of maps M_0,M_1,...,M_h such that $M_l = M_{l-1} \setminus K_{l-1}$ where K_{l-1} is a removal kernel in M_{l-1} for $l \in \{1, \dots, h\}$. In the remaining of the paper, we denote by $M_l = (\mathcal{D}_l, \gamma_{l,0}, \dots, \gamma_{l,n-1})$ the n-map of level l in a pyramid, for $0 \leq l \leq h$. We highlight the particular status of the base map by shortening $\gamma_{0,i}$ as γ_i for any dimension i.

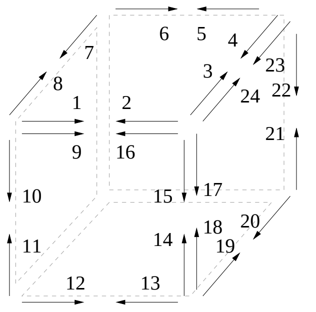

Figure 1. A 3-map which forms a partition of $\mathbb{R}^3$ into two regions: the inside and the outside of a cube.

3. Top-down relationships between levels

In the framework of irregular pyramids, reduction windows associate removed nodes in one level with a surviving node of the level above. In the context of pyramids of combinatorial maps, an analogous notion is the one of *connecting walk*. Indeed, we may associate with any dart of a map M_l a sequence of darts of M_{l-1} that must be traversed to define the new image of a dart of $\mathcal{D}_l$ by a permutation of M_l. Such a sequence of darts of M_{l-1} corresponds to a single dart in M_l according to the considered permutation.

Thus, for any $i \in \{0, \dots, n-1\}$, the *i-connecting walk* associated to a dart $d \in \mathcal{D}_l$, denoted by $\mathrm{CW}^i_l(d)$, is the sequence $(d_0 = d, d_1, \dots, d_p)$ where d_u is defined as follows for all $u \in \{0, \dots, p\}$:

- $d_u = d(\gamma_{l-1,i}\gamma_{l-1,i+1}^{-1})^u$, if $i \in \{0, \dots, n-2\}$.
- $d_u = d\gamma_{l-1,n-1}^u$, if $i = n-1$.

and $p = Min\{k \in \mathbb{N} \mid d_k\gamma_i \in \mathcal{D}'\}$.

Not surprisingly, this definition strictly follows the one of the removal operation. It is therefore readily seen that $d_u \notin \mathcal{D}_l$ if and only if $u > 0$, and we have proved [3, Property 5] that any such dart actually belongs to an i-cell of $\mathcal{D}_l$ (so does $d_u\gamma_i$ for $u \in \{0, \dots, p-1\}$).

It is also straightforward to prove that, if we denote $d_p = last(\mathrm{CW}^i_l(d))$, a connecting walk satisfies:

$$last(\mathrm{CW}^i_l(d))\gamma_{l-1,i} = d\gamma_{l,i} \qquad \text{[2, Prop. 6]}$$

Connecting walks may thus be used to compute the permutations $\gamma_{l,i}$ of M_l. Furthermore, since all connecting walks are proved to be disjoint [2, Proposition 3], this computation may be achieved with a sequential process in $\mathcal{O}(|\mathcal{D}_{l-1}|)$. The overall cost of computing M_l from M_{l-1} is thus equal to $\mathcal{O}(n|\mathcal{D}_{l-1}|)$.

Since a connecting walk associates to a dart d of level l a sequence of darts that have been removed from level $l-1$, we may define using the transtive closure of this relation a sequence of darts starting with d, but within the base level of the pyramid. This leads to the notion of *connecting dart sequence* which establishes a link, as shown by two propositions given further on, between any two levels of a pyramid the same way a connecting walk does between two consecutive levels.

Definition 1 (Connecting dart sequence [2]) *Let d be a dart of $\mathcal{D}_l$, $0 \leq l \leq h$. If $\mathrm{CW}^i_{M_{l-1},M_l}(d) = (d = d_0, \dots, d_p)$ for $i \in \{0, \dots, n-1\}$, the* i-connecting dart sequence *associated to d at level l, denoted by $\mathrm{CDS}^i_l(d)$, is defined by $\mathrm{CDS}^i_0(d) = (d)$ and, if $l > 0$, as follows:*

$$\mathrm{CDS}^i_l(d) = \mathrm{GL}^i_{l-1}(d_0) \cdot \ldots \cdot \mathrm{GL}^i_{l-1}(d_p)$$

where

$$\mathrm{GL}^i_{l-1}(d_r) = \begin{cases} \mathrm{CDS}^i_{l-1}(d_r) \cdot \mathrm{CDS}^{i+1}_{l-1}(d_r\gamma_{l-1,i}) \text{ if } r < p \\ \mathrm{CDS}^i_{l-1}(d_p) \text{ if } r = p \text{ or } i = n-1 \end{cases}$$

Important properties of the thus defined sequence of darts have been proved [2], among which an important one which states that for any dart d of $\mathcal{D}_l$, the image by $\gamma_{l,i}$ of d is precisely the image by $\gamma_{0,i}$ of the last dart of the connecting dart sequence $\mathrm{CDS}^i_l(d)$. In other words, we have

$$\mathit{last}(\mathrm{CDS}^i_l(d))\gamma_{0,i} = d\gamma_{l,i} \tag{1}$$

Together with a non-intersecting property, this shows that connecting dart sequences play a role similar to the one of receptive fields of the irregular pyramids framework, but in terms of darts rather than in terms of n-cells. One should also note that d is the only dart of $\mathrm{CDS}^i_l(d)$ belonging to $\mathcal{D}_l$.

Furthermore, the knowledge of the connecting dart sequence associated with a dart d immediately provides the value of $d\gamma_{l,i}$ using (1). However, the computation of $\mathrm{CDS}^i_l(d)$ following this very definition would require an explicit storage of all the maps under M_l, which therefore provides no space optimization.

4. Pyramid folding

The main contribution of this paper comes from the following proposition, which states that the connecting dart sequence $\mathrm{CDS}^i_l(d)$ may be retrieved using an iterative process. In this proposition, we use $\mathrm{AR}_{l,u}$ to denote the set of darts that belong to a u-cell of a removal kernel $K_{l'}$ for $l' < l$ (in particular, $\mathrm{AR}_{l,u} \cap \mathcal{D}_l = \emptyset$).

Proposition 1 *If* $\mathrm{CDS}^i_l(d) = (d_0, d_1, \ldots, d_p)$, $p \in \mathbb{N}^*$, *is the i-connecting dart sequence associated to a dart d of M_l, with $1 \leq l \leq h$ and $0 \leq i \leq n$; then for all $u \in \{0, \ldots, p-1\}$ we have $d_{u+1} = d_u\gamma_{t_u}$ with $t_0 = i$ and for all $u \in \{1, \ldots, p-1\}$ the subscript t_u satisfies:*

$$t_u = \begin{cases} t_{u-1} + 1 & \text{if } t_{u-1} < n-1 \text{ and } d_u \in \mathrm{AR}_{l,t_{u-1}} \\ t_{u-1} & \text{if } t_{u-1} = n-1 \text{ and } d_u \in \mathrm{AR}_{l,t_{u-1}} \\ t_{u-1} - 1 & \text{if } d_u \in \mathrm{AR}_{l,t_{u-1}-1} \end{cases} \tag{2}$$

Moreover $t_{p-1} = n-1$ if $i = n-1$, otherwise $t_{p-1} = i+1$. (It is proved that $p \neq 0$ implies that $p > 1$.)

The above property indeed provides an iterative definition of $\mathrm{CDS}^i_l(d)$. The sequence may thus be retrieved after exactly p decisions based on the position of the latter computed dart of the sequence relatively to sets of the form $\mathrm{AR}_{l,t}$, for some level l and dimension t. This position may be determined in constant time as soon as the following data is given for each dart d of the base map:

- $\Lambda(d)$, the level of the removal kernel that contains d, if any (otherwise we set $\Lambda(d) \overset{def}{=} h$); and
- $\Delta(d)$, the dimension of the cell that contains d within $K_{\Lambda(d)}$ (if $\Lambda(d) = h$ we set $\Delta(d) \overset{def}{=} n$, as no n-cell may be removed).

Indeed, for any dart $d \in \mathcal{D}_l$, we have:

$$d \in \mathrm{AR}_{l,t} \Leftrightarrow \Lambda(d) < l \wedge \Delta(d) = t$$

The above mentioned process may then traverse all the darts of the sequence, moving from one dart to its successor by applying a bounded number of permutations, based on a choice made in constant time depending on the two integers associated with any dart of the base map.

Using Proposition 1 as a mean to retrieve the connecting dart sequence of any dimension associated with a dart $d \in \mathcal{D}_l$, $l \leq h$, and using (1), we obtain that the image of d by any permutation of the map M_l may be computed using the functions Λ and Δ. It is therefore immediate that any map of the pyramid may be rebuilt efficientely (Section 3) given the base map and the two functions Λ and Δ. Eventually, we obtain an *implict encoding* of a whole pyramid as the base map and two functions. We may summarize this result as

$$(M_0, M_1, \ldots, M_h) \Leftrightarrow (M_0, \Lambda, \Delta)$$

The ($\Rightarrow$) part of this equivalence is straightforward from the very definition of the function Λ and Δ. The ($\Leftarrow$) part is a conséquence of Proposition 1 and (1).

4.1. Unfolding the pyramid

Given a triple (M_0, Λ, Δ), unfolding the pyramid consists in computing the values of the permutations $\gamma_0, \ldots, \gamma_{n-1}$ for all the darts of each level. Note that a set $\mathcal{D}_l$, $0 \leq l \leq h$, is determined by Λ using the relation $\mathcal{D}_l = \{d \in \mathcal{D}_0 \mid \Lambda(d) \geq l\}$. Eventually, the image by a permutation γ_i, $0 \leq i \leq n-1$, of any dart $d \in \mathcal{D}_l$ may be computed using Algorithm 1. Indeed, the while loop of the algorithm traverses the non-surviving darts of the connecting dart sequence while the *if* tests encode the three cases of Proposition 1 which are proved to be exclusive [2]. Furthermore, since any two connecting walks are distinct, it is readily seen that the same property holds for connecting dart sequences from their very définition. It follows that the unfolding of a whole level of the pyramid may be achieved in $\mathcal{O}(|\mathcal{D}|)$.

Algorithm 1: { Computes $\gamma_{l,i}(d)$ for $d \in \mathcal{D}_l$ }

```
Input: $M_0$, $\Lambda$ and $\Delta$
Input: $l \in \{1, \ldots, h\}$ {A level in the pyramid.}
Input: $i \in \{0, \ldots, n-1\}$ {A dimension.}
Input: $b \in \{1, \ldots, m\}$ with $\Lambda(d) \geq l$ {$b \in \mathcal{D}_l$.}
Output: $\gamma_{l,i}(d)$
$t_{prev} \leftarrow i$
$d \leftarrow b\gamma_i$
while $\Lambda(d) < l$ do
    if $\Delta(d) = t_{prev} - 1$ then
        $t_{prev} = t_{prev} - 1$
    else if $t_{prev} < n - 1$ then
        $t_{prev} \leftarrow t_{prev} + 1$
    $d \leftarrow d\gamma_{t_{prev}}$
return $d$
```

4.2. Space complexities

Let us now describe the benefits of the implicit encoding presented in the previous section in terms of space complexity.

An n-dimensionnal map with $N = |\mathcal{D}|$ darts may be stored using $n-1$ arrays of N integers each, that is $(n-1) \times N \times \log_2(N)$ bits. Assuming a constant decimation factor of τ between any two level of the pyramid, the number of darts in the map M_l, $0 \leq l \leq h$ is N/τ^l so that the overall number of bits used to store the pyramid explicitely is

$$(n-1)\log_2(N) \sum_{l=0}^{h} \frac{N}{\tau^l} \simeq (n-1)N\log_2(N)\frac{\tau}{\tau-1}$$

Given $N = |\mathcal{D}_0|$, when h is maximal we have $N/\tau^h = 1$, hence $h\log_2(\tau) = \log_2(N)$. In this case, the size becomes $h(n-1)N\tau\log_2(\tau)/(\tau-1)$ bits. With $\tau = 2$ we obtain $2hN(n-1)$ which grows in a linear way with respect to the height h of the pyramid.

On the other hand, the implicit encoding requires the storage of the base maps and exactly two integers for each $d \in \mathcal{D}_0$: the dimension $\Delta(d)$, bounded by n, and the level $\Lambda(d)$, bounded by h. In other words, the implicit encoding of a pyramid with height h requires

$$(n-1)\log_2(N) + N \times (\log_2(n) + \log_2(h)) \text{ bits.}$$

This quantity therefore grows as $\mathcal{O}(\log_2(h))$.

Considering a base n-map that partitions an hypercube with side $m \in \mathbb{N}$ in the cubic grid, the number of darts $C_n(m)$ in such a map is given by the relation:

$$\begin{aligned} C_1(m) &= 2m, \text{ and} \\ C_n(m) &= (m^n \times 2^{n-1} \times n!) + n \times C_{n-1}(m) \end{aligned}$$

As an illustration, the table below summarizes actual figures obtained for storing pyramids of 3D combinatorial maps in the cubic grid, with $\tau = 2$.

Side length	Height	Explicit enc.	Implicit enc.
512	32	48 GiB	2.6 GiB
1024	35	420 GiB	24 GiB

Because the size of an implicit encoding has a small dependance with respect to the height of the pyramid, it may permit during a union-only process the merging of only a few regions between consecutive levels of the pyramid (i.e. using a small decimation factor), which therefore should preserve subsequent mergings to be too dependent on the choices made during previous steps.

Furthermore, as the complexity of Algorithm 1 is linear with the length of $\mathrm{CDS}_l^i(d)$, the computation of the values of $\gamma_{l,i}$ for all the darts of $\mathcal{D}_l$ is therefore in $\mathcal{O}(N)$, hence the unfolding of any level of the pyramid may be achieved in $\mathcal{O}(N \times n)$. Eventually, a parallel implementation with one processor per dart of $\mathcal{D}_l$ would be in linear time with respect to the dimension.

5. Conclusion

We have defined an implicit encoding of n-dimensional combinatorial pyramids (see [1]) the way Brun and Kropatsch did in the two-dimensional case ([1]) and following the works of Grasset et al. about pyramids of generalized maps ([5]). This encoding relies on the definitions of connecting walks (*reduction windows*) and connecting dart sequences (*receptive fields*).

References

[1] L. Brun and W. Kropatsch. Combinatorial pyramids. In Suvisoft, editor, *IEEE International conference on Image Processing (ICIP)*, volume II, pages 33–37, Barcelona, September 2003. IEEE.

[2] S. Fourey and L. Brun. Connecting walks and connecting dart sequences for n-D combinatorial pyramids. In P. Wiederhold and R. P. Barneva, editors, *Progress in Combinatorial Image Analysis (International Workshop on Combinatorial Image Analysis)*, pages 109–122, Cancun, Mexico, Nov. 2009. Research Publishing Services.

[3] S. Fourey and L. Brun. Connecting walks and connecting dart sequences in nD combinatorial pyramids. Technical report TR-2009-02, GREYC, 2009. `http://hal.archives-ouvertes.fr/?langue=en`.

[4] S. Fourey and L. Brun. A first step toward combinatorial pyramids in nD spaces. In *Graph-Based Representations in Pattern Recognition*, volume 5534 of *Lecture Notes in Computer Science*, pages 304–313. Springer, May 2009.

[5] C. Grasset-Simon, G. Damiand, and P. Lienhardt. *n*D generalized map pyramids: Definition, representations and basic operations. *Pattern Recognition*, 39(4):527–538, 2006.

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

6. Space complexity with n-D cubic grids

In the cubical grid, an n-cell contains exactly $2n$ faces (i.e. $(n-1)$-cells). From the very definition of cells in n-maps as sets of darts, any two $(n-1)$-cells are disjoint. We thus obtain that c_n, the number of darts in an n-cell, is obtained by the relation

$$c_n = 2 \times n \times f$$

where f is the number of darts in an $(n-1)$-cell.

In fact, an $(n-1)$-cell is nothing but an $(n-1)$-map, in the $(n-1)$-D cubic grid, that separates the space in two cells: an $(n-1)$-unit cube and the surrounding unbounded cell. It follows that $f = c_{n-1}$, so that

$$c_n = 2 \times n \times c_{n-1}$$

Since $c_1 = 2$ we obtain:

$$c_n = 2^n \times n!$$

If we consider an n-map that represents an n-D cube with edge length m and sum up the number of darts as $m^n \times c_n$, then all the darts of the inner faces of the cube are counted twice as an $(n-1)$-cell belongs to two n-cells in this case.

On the other hand, darts of faces that constitute the sides of the hypercube are shared between a unit n-cell of the cube and the outer infinite n-cell surrounding the cube. We therefore may compute the total number of darts in such a map, denoted by $C_n(m)$ as

$$C_n(m) = \frac{m^n \times c_n}{2} + n \times C_{n-1}(m) \qquad (3)$$

Indeed, sides of the m^n-cube are $(n-1)$-maps representing m^{n-1} cubes, and an m^n-cube has $2n$ sides. The sides of the cube thus contain $2 \times n \times C_{n-1}(m)$. However, since the first term of (4) already counts half of the darts in these sides of the cube, we should divide this second term by two.

Eventually, we have:

$$\begin{aligned} C_1(m) &= 2m \\ C_n(m) &= (m^n \times 2^{n-1} \times n!) + n \times C_{n-1}(m) \end{aligned}$$

And, with $m = 3$:

$$\begin{aligned} C_1(3) &= 6 \\ C_2(3) &= 3^2 \times 2^1 \times 2! + 2 \times 6 = 48 \\ C_3(3) &= 3^3 \times 2^2 \times 3! + 3 \times 48 = 792 \\ C_4(3) &= 3^4 \times 2^3 \times 4! + 4 \times 792 = 18720 \end{aligned}$$