Receptive fields within the Combinatorial Pyramid framework

Luc Brun† and Walter Kropatsch⋆‡

† Laboratoire d'Études et de Recherche en Informatique(EA 2618)
Université de Reims - France
and
‡ Institute for Computer-aided Automation
Pattern Recognition and Image Processing Group
Vienna Univ. of Technology- Austria

† brun@leri.univ-reims.fr, ‡ krw@prip.tuwien.ac.at

Abstract. A hierarchical structure is a stack of successively reduced image representations. Each basic element of a hierarchical structure is the father of a set of elements in the level below. The transitive closure of this father-child relationship associates to each element of the hierarchy a set of basic elements in the base level image representation. Such a set, called a receptive field, defines the embedding of one element of the hierarchy on the original image. Using the father-child relationship, global properties of a receptive field may be computed in $\mathcal{O}(\log(m))$ parallel processing steps where m is the diameter of the receptive field. Combinatorial pyramids are defined as a stack of successively reduced combinatorial maps, each combinatorial map being defined by two permutations acting on a set of half edges named darts. The basic element of a combinatorial pyramid is thus the dart. This paper defines the receptive field of each dart within a combinatorial pyramid and study the main properties of these sets.

1 Introduction

Regular image pyramids have been introduced 1981/82 [11] as a stack of images with exponentially reduced resolution. Each image of this sequence is called a *level*. Such Pyramids present several interesting properties within the image processing and analysis framework such as [4]: The reduction of noise, the processing of local and global features within the same frame and the efficiency of many computations on this structure. Using the neighborhood relationships defined on each image the *Reduction window* relates each pixel of the pyramid with a set of pixels defined in the level below. The pixels belonging to one reduction window are the *children* of the pixel which defines it. This father-child relationship maybe extended by transitivity down to the base level image. **The set of**

⋆ The authors wish to thank the anonymous reviewers for their useful comments. This Work was supported by the Austrian Science Foundation under P14445-MAT.

children of one pixel in the base level is named its *receptive field* (RF) and defines the embedding of this pixel on the original image. Using the father-child relationship global properties of a receptive field $RF(v)$ with a diameter m may be computed in $\mathcal{O}(\log(m))$ parallel processing steps thanks to local calculus. However, receptive fields defined within the regular pyramid framework are not necessarily connected [4]. Furthermore, the adjacency of two pixels v and w defined at level k may not be easily interpreted on the base level image. Indeed, the boundary between the receptive fields $RF(v)$ and $RF(w)$ associated to this adjacency at level k may be disconnected and even incomplete.

Irregular pyramids, first introduced by Meer [16], Montanvert [17] and Jolion [13] are defined as a stack of successively reduced simple graphs (i.e. graphs without double edges nor self-loops). The base level graph may be built from a sampling grid using one pixel adjacency such as the $4-$neighborhood. Each graph of the hierarchy is built from the graph below by selecting a set of vertices named surviving vertices and mapping each non surviving vertex to a surviving one [17]. This mapping induces a father-child relationship between a surviving vertex and the set of non surviving vertices mapped to it. The reduction window of one surviving vertex is then defined as its set of children. The receptive field of one surviving vertex is defined by the transitive closure of the father-child relationship. Using this reduction scheme, the receptive field of each vertex in the hierarchy is a **connected set of vertices in the base level graph**. However, using simple graphs, the adjacency between two vertices is encoded by only one edge while the receptive fields of two vertices may share several boundaries. Edges in the hierarchy may thus encode non connected set of boundaries between the associated receptive fields. Moreover, the lack of self-loops in simple graphs does not allow to differentiate an adjacency relationship between two receptive fields from an inclusion relationship.

The last two drawbacks may be overcome by using the Dual graph pyramids introduced by Kropatsch [14]. Using Kropatsch's reduction scheme, the reduction operation is encoded by edge contractions [14]. This operation contracts one edge and its two end points into a single vertex. The contraction of a graph reduces the number of vertices while maintaining the connections to other vertices. As a consequence some redundant edges such as self-loops or double edges may occur. These redundant edges may be characterized in the dual of the contracted graph. The removal of such edges is called a dual decimation step. Since the reduction scheme requires both features of the initial graph and its dual such pyramids are called *Dual graph pyramids*. Within such hierarchies, each receptive field is a **connected set of vertices in the base level**. Moreover, each edge between two vertices in the hierarchy encodes an unique connected boundary between the associated receptive fields. Finally, the use of self-loops within the hierarchy allows to differentiate adjacency relationships between receptive fields from inclusions relations.

The basic entity of all the above pyramids is the vertex/pixel. Combinatorial maps are based on darts. Hence **receptive fields of Combinatorial Pyramids are expressed in terms of darts**. Combinatorial pyramids are equivalent to

dual graph pyramids with the exeption that they represent the orientation explicitely. The expected advantages of such hierarchies within the image analysis framework are presented in [9].

The remaining of this paper is as follows: In Section 2 we present the combinatorial map model together with the expected advantages of this model within the Pyramid framework. In Section 3 we present the construction scheme of a combinatorial pyramid. Finally Section 4 defines the notion of receptive field within the combinatorial pyramid framework and states its major properties.

2 Combinatorial maps

Combinatorial maps and generalized combinatorial maps define a general framework which allows to encode any subdivision of nD topological spaces orientable or non-orientable with or without boundaries. The concept of maps has been first introduced by Edmonds [12] in 1960 and later extended by several authors [15]. This model has been applied to several fields of computer imagery such as geometrical modeling [3] and 2D segmentation [1, 5]. An exhaustive comparison of combinatorial maps with other boundary representations such as cell-tuples and quad-edges is presented in [15]. Recent trends in combinatorial maps apply this framework to the segmentation of 3D images [6, 2] and the encoding of hierarchies [8, 9].

The remaining of this paper will be based on 2D combinatorial maps which will be just called combinatorial maps. A combinatorial map may be seen as a planar graph encoding explicitly the orientation of edges around a given vertex. Fig. 1a) demonstrates the derivation of a combinatorial map from a plane graph. First edges are split into two half edges called *darts*, each dart having its origin at the vertex it is attached to. The fact that two half-edges (darts) stem from the same edge is recorded in the reverse permutation α. A second permutation σ encodes the set of darts encountered when turning counterclockwise around a vertex (see e.g. the σ-orbit $(-8, -3, 11, 4)$ encoding the central vertex in Fig. 1a)).

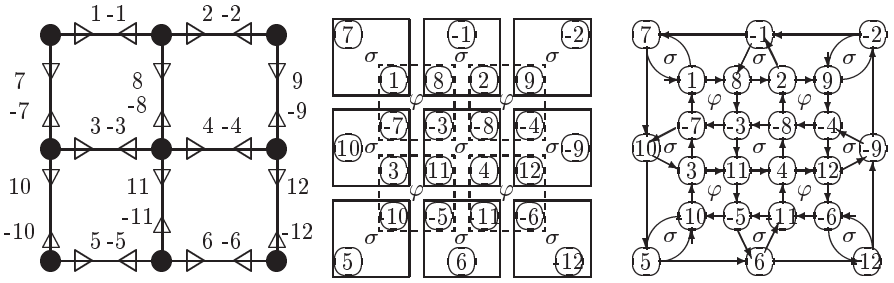

Fig. 1. A 3×3 grid encoded by a combinatorial map

The symbols $\alpha^*(d)$ and $\sigma^*(d)$ stand, respectively, for the α and σ orbits of the dart d. More generally, if d is a dart and π a permutation we will denote the π-orbit of d by $\pi^*(d)$.

A combinatorial map G is the triplet $G = (\boldsymbol{\mathcal{D}}, \sigma, \alpha)$, where $\boldsymbol{\mathcal{D}}$ is the set of darts and σ, α are two permutations defined on $\boldsymbol{\mathcal{D}}$ such that α is an involution:

$$\forall d \in \boldsymbol{\mathcal{D}} \quad \alpha^2(d) = d \tag{1}$$

Note that, if the darts are encoded by positive and negative integers, the involution α may be implicitly encoded by sign (Fig. 1a)). This convention is often used for practical implementations [5] where the combinatorial map is simply implemented by an array of integers encoding the permutation σ.

Given a combinatorial map $G = (\boldsymbol{\mathcal{D}}, \sigma, \alpha)$, its dual is defined by $\overline{G} = (\boldsymbol{\mathcal{D}}, \varphi, \alpha)$ with $\varphi = \sigma \circ \alpha$. The orbits of the permutation φ encode the set of darts encountered when turning around a face (see e.g. the φ-orbit $(1, 8, -3, -7)$ in Fig. 1a)). Note that, using a counter-clockwise orientation for permutation σ, each dart of a φ-orbit has its associated face on its right.

Figures 1b) and 1c) illustrate an alternative representation of the combinatorial map encoding. Within such a representation, each dart is represented by one vertex and one edge connects a dart d_1 to d_2 iff either $d_2 = \sigma(d_1)$ or $d_2 = \varphi(d_1)$. Using this representation, the σ and φ orbits of the combinatorial map are represented by the faces of the oriented graph.

3 Combinatorial Pyramids

The aim of combinatorial pyramids is to combine the advantages of combinatorial maps with the reduction scheme defined by Kropatsch [14] (see also Section 1). A combinatorial pyramid is thus defined by an initial combinatorial map successively reduced by a sequence of contraction or removal operations. In order to preserve the number of connected components of the initial combinatorial map, we forbid the removal of bridges and the contraction of self-loops. A self-loop in the initial combinatorial map becomes a bridge in its dual and vice-versa [10]. In the same way, a contraction operation in the initial combinatorial map is equivalent to a removal operation performed in its dual. Therefore, the exclusion of bridges and self-loops from respectively removal and contraction operations corresponds to a same constraint applied alternatively on the dual combinatorial map and the original one.

In order to avoid the contraction of self-loops, the set of edges to be contracted must form a forest of the initial combinatorial map. The graph of a combinatorial map is a forest if it does not contain a cycle. A more formal definition may be found in [7][Def. 4]. A set of edges to be contracted satisfying the above requirement is called a contraction kernel:

Definition 1. Contraction Kernel

Given a connected combinatorial map $G = (\boldsymbol{\mathcal{D}}, \sigma, \alpha)$ the set $K \subset \boldsymbol{\mathcal{D}}$ will be called a contraction kernel iff K is a forest of G.

The set $\mathcal{SD} = \boldsymbol{\mathcal{D}} - K$ is called the set of surviving darts.

In the same way, a removal kernel is defined as a forest of the dual combinatorial map. This constraint insures that no self-loop will be contracted in the dual combinatorial map and thus that no bridge may be removed in the initial one. Within our framework, a removal kernel is used to remove redundant edges created by contraction operations. These edges are characterized as self-loops or double edges [14]. A removal kernel is thus defined as a forest of the dual combinatorial map removing any redundant double edge and self-loop.

Since one contraction encodes a merge between two regions, the set of darts encoding the adjacency relationships between the image and its background must be excluded from contraction operations. However, after a contraction step, some edges of the image boundary may become double edges. The removal of such edges corresponds to the concatenation of the associated boundaries in the base level image. Such operation preserves the image boundary and is thus allowed.

Contraction and removal kernels specify the set of edges which must be contracted or removed. The creation of the reduced combinatorial map from a contraction or a removal kernel is performed in parallel by using connecting walks [9]. Given a combinatorial map $G = (\boldsymbol{\mathcal{D}}, \sigma, \alpha)$, a kernel K and a surviving dart $d \in \mathcal{SD} = \boldsymbol{\mathcal{D}} - K$, the connecting walk associated to d is either equal to:

$$CW(d) = d, \varphi(d), \ldots, \varphi^{n-1}(d) \text{ with } n = Min\{p \in \mathbb{N}^* \mid \varphi^p(d) \in \mathcal{SD}\}$$

if K is a contraction kernel and

$$CW(d) = d, \sigma(d), \ldots, \sigma^{n-1}(d) \text{ with } n = Min\{p \in \mathbb{N}^* \mid \sigma^p(d) \in \mathcal{SD}\}$$

If K is a removal kernel.

Given a kernel K and a surviving dart $d \in \mathcal{SD}$, such that $CW(d) = d.d_1 \ldots d_p$, the successor of d within the reduced combinatorial map $G' = G/K = (\mathcal{SD}, \sigma', \alpha)$ is retrieved from $CW(d)$ by the following equations [9]:

$$\begin{array}{l} \varphi'(d) = \varphi(d_p) \text{ if } K \text{ is a contraction kernel} \\ \sigma'(d) = \sigma(d_p) \text{ if } K \text{ is a removal kernel} \end{array} \quad (2)$$

Note that, if K is a contraction kernel, the connecting walk $CW(d)$ allows to compute $\varphi'(d)$. The σ-successor of d within the contracted combinatorial maps may be retrieved from $CW(\alpha(d)) = \alpha(d).d'_1, \ldots, d'_p$. Indeed, we obtain by using equations 1 and 2: $\varphi'(\alpha(d)) = \sigma'(\alpha(\alpha(d))) = \sigma'(d) = \varphi(d'_p)$.

Fig. 2a) shows the three connection walks defined by kernel $\alpha^*(1, 7, 10)$ corresponding to the contracted maps in Fig. 2b). On this example, we have $\varphi'(-2) = 5, \varphi'(-5) = 3$ and $\varphi'(-3) = 8$. The permutation φ' remains unchanged for the other surviving darts. We have thus: $\sigma'(2) = 5, \sigma'(5) = 3$ and $\sigma'(3) = 8$ while the permutation σ' is unchanged for the other surviving darts.

Based on the above property we designed two algorithms to traverse connecting walks respectively defined by contraction and removal kernels [9]. One straightforward application of these algorithms consists to compute the reduced combinatorial maps. However, connecting walks may also be used to relate one combinatorial map of the pyramid with the one below. Indeed, we showed [8] that

each non surviving dart belongs to one and only one connecting walk. Therefore, the connecting walk $CW(d)$ of each surviving dart d contains the dart d and all non surviving darts mapped to it. Such a sequence of darts corresponds to the reduction window associated to the surviving dart d.

4 Receptive fields of darts

The notion of connecting walks introduced in Section 3 allows us to build one reduced combinatorial map from an initial one and a contraction or removal kernel. Therefore, given a sequence of kernels $K_1, \ldots, K_n$ and an initial combinatorial map $G_0 = (\mathcal{D}, \sigma, \alpha)$ defined from a planar sampling grid (e.g. the $4-$neighborhood one) one can define the sequence of reduced combinatorial maps $G_0, G_1, \ldots, G_n$ where $G_i = G_{i-1}/K_i = (\mathcal{SD}_i, \sigma_i, \alpha)$ for each $i \in \{1, \ldots, n\}$. Note that according to the definition of surviving darts (Definition 1) we have $\mathcal{SD}_i = \mathcal{SD}_{i-1} - K_i = \mathcal{D} - \cup_{j=1}^{i} K_j$.

Intuitively, one connecting walk $CW_i(d)$ defines the set of darts that we have to traverse in the level below in order to connect d to $\varphi_i(d)$ if K_i is a contraction kernel and d to $\sigma_i(d)$ if K_i is a removal kernel (see equation 2). Let us consider the sequence of darts $CDS_i(d)$ that we have to traverse in the base level graph G_0 to connect d to $\varphi_i(d)$ if K_i is a contraction kernel and d to $\sigma_i(d)$ if K_i is a removal kernel. Such a sequence of darts is called a connecting dart sequence(CDS). Moreover, using the construction scheme described below, we showed [8] that the first dart of $CDS_i(d)$ is d. A connecting dart sequence $CDS_i(d)$ without its first dart will be denoted $CDS_i^*(d)$.

To construct a CDS, let us first suppose that K_i is a contraction kernel. If K_{i+1} is a removal kernel we have to traverse the sequence of darts $CW_{i+1}(d) = d.d_1, \ldots, d_p$ in G_i to connect d to $\sigma_{i+1}(d)$. Each dart of $CW_{i+1}(d)$ is related to the following one by:

$$d_1 = \sigma_i(d),\ \forall j \in \{1, \ldots, p-1\} \quad d_{j+1} = \sigma_i(d_j)$$

Since K_i is a contraction kernel, $CDS_i(d_j)$ connects d_j to $\varphi_i(d_j)$ for any j in $\{1, \ldots, p-1\}$. However, $CDS_i(\alpha(d_j))$ connects $\alpha(d_j)$ to $\varphi_i(\alpha(d_j)) = \sigma_i(\alpha \circ \alpha(d_j)) = \sigma_i(d_j)$ (see equation 1). The connection between d_j and $\sigma_i(d_j)$ is thus performed by $d_j CDS_i^*(\alpha(d_j))$ and the connecting dart sequence of d at level $i+1$ is equal to $CDS_{i+1}(d) = d_1 \cdot CDS_i^*(\alpha(d_1)) \cdots d_p \cdot CDS_i^*(\alpha(d_p))$.

Fig. 2c) illustrates the connecting dart sequences defined by the applications of the contraction kernel $K_1 = \alpha^*(1, 7, 10)$ followed by the removal kernel $K_2 = \alpha^*(3)$. Connecting walks defined by K_1 are illustrated in Fig. 2a) while the contracted combinatorial map $G_1 = G_0/K_1$ is represented in Fig. 2b) together with the connecting walks defined by K_2. According to Fig. 2b) we have to traverse the dart 3 in G_1 to connect 5 to $\sigma_2(5) = 8$. Moreover, using Fig. 2a), the connection between 5 and 3 requires to traverse the dart -10 in G_0 while the connection between 3 and 8 requires to traverse the darts -7 and 1. The connecting dart sequence associated to 5 is thus equal to $CDS_2(5) = 5CDS_1^*(-5)3.CDS_1^*(-3) = 5.-10.3.-7.1$ (Fig. 2c)).

Conversely, if both kernels K_i and K_{i+1} are contraction kernels, the connecting dart sequence of d at level $i+1$ is equal to [8]: $CDS_i(d).CDS_i(d_1)\cdots CDS_i(d_p)$. The same construction scheme holds if K_i is a removal kernel. Both cases are resumed in the following definition:

Definition 2. *Given a combinatorial map $G_0 = (\mathcal{D}, \sigma, \alpha)$ and a sequence of contraction or removal kernels $K_1, K_2 \ldots, K_n$. The connecting dart sequences are defined by the following recursive construction:*

$$\forall d \in \mathcal{D} \quad CDS_0(d) = d$$

For each level i in $\{1, \ldots, n\}$ and for each dart d in $\mathcal{SD}_i$

- *If K_i and K_{i-1} have the same type:*

$$CDS_i(d) = CDS_{i-1}(d_1)\cdots CDS_{i-1}(d_p)$$

- *If K_i and K_{i-1} have different types:*

$$CDS_i(d) = d_1 \cdot CDS^*_{i-1}(\alpha(d_1))\cdots d_p \cdot CDS^*_{i-1}(\alpha(d_p))$$

Where $(d_1 \ldots d_p)$ is equal to $CW_i(d)$. The kernels $K_0 = \emptyset$ and K_1 have the same type by convention.

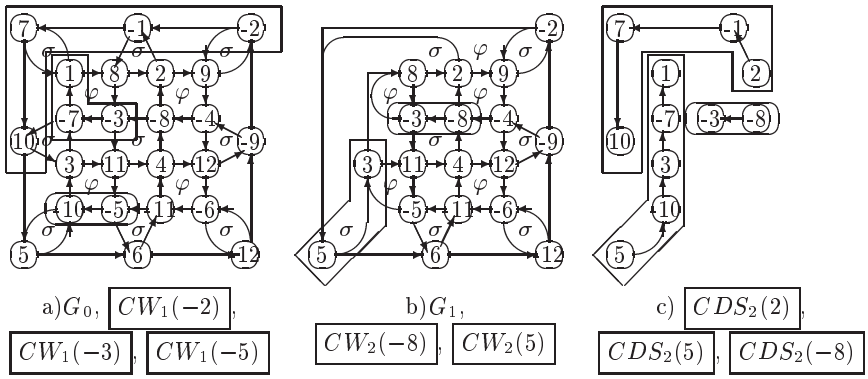

Fig. 2. Receptive fields CDS_2 are based on connecting walks CW_1 and CW_2.

The set of connecting dart sequences defined at each level defines a partition of the initial set of darts $\mathcal{D}$. Moreover, each connecting dart sequence $CDS_i(d)$ defined at level i by $d \in \mathcal{SD}_i$ satisfies [8] $CDS^*_i(d) \subset \cup_{j=1}^{i} K_i$. Therefore, d is the only dart of $CDS_i(d)$ surviving up to level i and $CDS^*_i(d)$ encodes the set of non surviving darts mapped to it at level i. Connecting dart sequences encode thus the notion of receptive field within the combinatorial map framework.

Using Definition 2 connecting dart sequences must be computed at each level from the level below. However, such a construction scheme may induce useless calculus if one does not need to compute all connecting dart sequences defined from level 1 to level i. This recursive construction scheme may be avoided by using the following theorem [8]:

Theorem 1 *Given a combinatorial map $G_0 = (\mathcal{D}, \sigma, \alpha)$, a sequence of contraction kernels or removal kernels $K_1, K_2 \ldots, K_n$, the relation between the successive darts of a connecting dart sequence $CDS_i(d) = (d.d_1, \ldots, d_{p-1})$, with $i \in \{1, \ldots, n\}$ and $d \in \mathcal{SD}_i$ is as follows:*

$$d_1 = \begin{cases} \varphi(d) \text{ If } K_i \text{ is a contraction kernel} \\ \sigma(d) \text{ If } K_i \text{ is a removal kernel} \end{cases}$$

$$\forall j \in \{2, \ldots, p\}\ d_j = \begin{cases} \varphi(d_{j-1}) \text{ if } d_{j-1} \text{ has been contracted} \\ \sigma(d_{j-1}) \text{ if } d_{j-1} \text{ has been removed} \end{cases}$$

Therefore, connecting dart sequences may be computed directly from the base level graph G_0 given the type of the kernel K_i and the type of the reduction operation applied to each dart in $\cup_{j=1}^{i} K_j$. This last information may be stored in each dart during the construction of the pyramid by a function *state* from $\cup_{j=1}^{i} K_j$ to $\{Contracted, Removed\}$ such as $state(d)$ is equal to either $Contracted$ or $Removed$ according to the type of the kernel applied to d before level i. Note that the storage of this function only adds on bit per dart.

Algorithm 1 uses the function *state* and the properties established by Theorem 1 to compute one connecting dart sequence at level i from the base level graph G_0. The complexity of this algorithm is linear in the length of the computed connecting dart sequence. Algorithms computing all connecting dart sequences defined at level i in parallel are described in [8]. The complexity of these algorithms is $\mathcal{O}(\log(M_i))$ where M_i denotes the longest connecting dart sequence defined at level i.

Moreover, by construction connecting dart sequences encode the set of darts in G_0 that we have to traverse to connect one surviving dart at level i with its φ_i or σ_i successor according to K_i. Indeed, given one dart $d \in \mathcal{SD}_i$, such that $CDS_i(d) = d.d_1, \ldots, d_p$ we showed [8] that:

- If K_i is a contraction kernel:

$$\varphi_i(d) = \begin{cases} \varphi(d_p) \text{ if } d_p \text{ has been contracted} \\ \sigma(d_p) \text{ if } d_p \text{ has been removed} \end{cases} \quad (3)$$

- If K_i is a removal kernel:

$$\sigma_i(d) = \begin{cases} \varphi(d_p) \text{ if } d_p \text{ has been contracted} \\ \sigma(d_p) \text{ if } d_p \text{ has been removed} \end{cases} \quad (4)$$

Note that equations 3 and 4 are similar to equations 2 defined for connecting walks. Given an encoding of the function *state* and the set of surviving darts $\mathcal{SD}_i$, equations 3 and 4 combined with Algorithm 1 allow us to retrieve the reduced

```
cds compute_cds(map G_0,dart d,level i)
{
    cds CDS=d
    if (K_i is a contraction kernel)
        compute_cds_rec(G_0,φ(d),i,CDS);
    else
        compute_cds_rec(G_0,σ(d),i,CDS);
    return CDS
}

cds compute_cds_rec(map G_0,dart d',level i,cds CDS)
{
    if (d' ∈ SD_i)
        return CDS;
    CDS=CDS.d'
    if (state(d') = Contracted)
        compute_cds_rec(G_0,φ(d'),i,CDS);
    else
        compute_cds_rec(G_0,σ(d'),i,CDS);
    return CDS
}
```

Algorithm 1: *Computation of the connecting dart sequence of a dart d at level i.*

combinatorial map G_i without computing $G_1, \ldots, G_{i-1}$. This last property may be useful for the analysis of hierarchies where the first levels of the pyramid often corresponds to low level operations.

5 Conclusion

The presented concept of connecting dart sequences uses the principle of receptive fields for the first time within the combinatorial pyramid framework. The darts of a combinatorial map refine the pixel subdivision of a digital image. Hence receptive fields based on darts can distinguish more configurations. Moreover, using the order defined on such sequences we showed that any reduced combinatorial map may be retrieved directely from the base level.

Higher level concepts are typically associated with higher level structural entities like a dart or a vertex at a high pyramid level. Using the receptive field of these structural entities properties and parameters of the corresponding high level concepts may be computed, maybe through complicated calculations, maybe using the hierarchical decomposition, from the gray values, the colors or the coordinates of the discrete (pixel) measurements. In the future we plan to study the interactions between the highly flexible contraction scheme of combinatorial pyramids and the efficiency of computing high level descriptions

References

[1] E. Ahronovitz, J. Aubert, and C. Fiorio. The star-topology: a topology for image analysis. In *5th DGCI Proceedings*, pages 107–116, 1995.

[2] Y. Bertrand, G. Damiand, and C. Fiorio. Topological map: Minimal encoding of 3d segmented images. In J. M. Jolion, W. Kropatsch, and M. Vento, editors, 3^{rd} *Workshop on Graph-based Representations in Pattern Recognition*, pages 64–73, Ischia(Italy), May 2001. IAPR-TC15, CUEN.

[3] Y. Bertrand and J. Dufourd. Algebraic specification of a 3D-modeler based on hypermaps. *CVGIP: Graphical Models and Image Processing*, 56(1):29–60, Jan. 1994.

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

[5] J. P. Braquelaire and L. Brun. Image segmentation with topological maps and inter-pixel representation. *Journal of Visual Communication and Image representation*, 9(1), 1998.

[6] J. P. Braquelaire, P. Desbarats, and J. P. Domenger. 3d split and merge with 3-maps. In J. M. Jolion, W. Kropatsch, and M. Vento, editors, 3^{rd} *Workshop on Graph-based Representations in Pattern Recognition*, pages 32–43, Ischia(Italy), May 2001. IAPR-TC15, CUEN.

[7] L. Brun and W. Kropatsch. Pyramids with combinatorial maps. Technical Report PRIP-TR-057, PRIP, TU Wien, 1999.

[8] L. Brun and W. Kropatsch. The construction of pyramids with combinatorial maps. Technical Report 63, Institute of Computer Aided Design, Vienna University of Technology, lstr. 3/1832,A-1040 Vienna AUSTRIA, June 2000.

[9] L. Brun and W. Kropatsch. Contraction kernels and combinatorial maps. In J. M. Jolion, W. Kropatsch, and M. Vento, editors, 3^{rd} *IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, pages 12–21, Ischia Italy, May 2001. IAPR-TC15, CUEN.

[10] L. Brun and W. G. Kropatsch. Dual contraction of combinatorial maps. In W. G. Kropatsch and J.-M. Jolion, editors, 2^{nd} *IAPR-TC-15 Workshop on Graph-based Representations*, volume 126, pages 145–154, Haindorf, Austria, May 1999. Österreichische Computer Gesellschaft.

[11] P. Burt, T.-H. Hong, and A. Rosenfeld. Segmentation and estimation of image region properties through cooperative hierarchial computation. *IEEE Transactions on Sustems, Man and Cybernetics*, 11(12):802–809, December 1981.

[12] J. Edmonds. A combinatorial representation for polyhedral surfaces. *Notices American Society*, 7, 1960.

[13] J. Jolion and A. Montanvert. The adaptative pyramid : A framework for 2d image analysis. *Computer Vision, Graphics, and Image Processing*, 55(3):339–348, May 1992.

[14] W. G. Kropatsch. Building Irregular Pyramids by Dual Graph Contraction. *IEE-Proc. Vision, Image and Signal Processing*, Vol. 142(No. 6):pp. 366–374, December 1995.

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

[16] P. Meer. Stochastic image pyramids. *Computer Vision Graphics Image Processing*, 45:269–294, 1989.

[17] A. Montanvert, P. Meer, and A. Rosenfeld. Hierarchical image analysis using irregular tessellations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):307–316, APRIL 1991.