Technical Report

Dept. for Pattern Recognition and Image Processing
Institute for Automation
Technical University of Vienna
Treitlstr. 3/1832
A-1040 Vienna AUSTRIA
Phone: +43 (1) 58801-8161
Fax: +43 (1) 586 96 97
E-mail: krw@prip.tuwien.ac.at

PRIP-TR-42 November 17, 1995

Equivalent Contraction Kernels and The Domain of Dual Irregular Pyramids[1]

Walter G. Kropatsch

Abstract

Dual graph contraction reduces the number of vertices and of edges of a pair of dual image graphs while, at the same time, the topological relations among the 'surviving' components are preserved. Repeated application produces a stack of successively smaller graphs: a pair of dual irregular pyramids. The process is controlled by selected decimation parameters which consist of a subset of surviving vertices and associated contraction kernels. Equivalent contraction kernels (ECKs) combine two or more contraction kernels into one single contraction kernel which generates the same result in one single dual contraction. Decimation parameters of any individual pyramid level can be reconstructed from the ECK of the pyramid's apex if both vertices and edges of this ECK receive labels indicating their annihilation level in the pyramid. This is a labeled spanning tree (LST) of the base graph which allows efficient design and control of different types of dual irregular pyramids. Since the LST determines the pyramid, primitive modifications of the LST transform also pyramids into other pyramids on the same base graph. They open a large variety of possibilities to explore the domain of 'all' pyramids.

[1]This work was supported by the Austrian Science Foundation under grant S 7002-MAT

1 Introduction

A raw digital image consists of a 2D spatial arragement of pixels each of which results from measuring the light at a specific location of the image plane. Currently most of the artificial sensors (e.g. CCD cameras) have the rigid structure of an orthogonal grid, whereas most natural vision systems are based on non-regular arrangements of sensors [1]. Although arrays are certainly easier to manage technically, topological relations seem to play an even more important role for vision tasks in natural systems than precise geometrical positions.

A second aspect concerns the projection from the real (3D-) world into the 2D image. Surfaces of 3D-objects reflect the light in a very specific way that somehow 'codes' the structure of the object: reflectivity within homogeneous regions does not vary much, it changes abruptly between different surfaces or from the object to its background [18]. The topological structure on a visible surface patch is preserved in the image while its geometry may be severly distorted. But also the arrangement of different objects in the 3D-world will be mapped to the regions in the image, be it regularly or irregularly sampled. Hence the idea pursuit in this paper to start with arbitrarily but densely sampled measurements of which only the topology is known and to successively shrink the number of descriptive elements until the structure of the imaged scene becomes evident.

The third aspect addresses computer vision models. They have in general a parametric and a structural component. While parameter optimization models quantitative image properties well, the qualitative image and scene properties rely more on the structural component.

The presented approach addresses a representation of pure structure, a hierarchy of plane graphs, with a clear interface, the decimation parameters, to control generation and modification of the structure. Dual graph contraction is the basic process [13] that builds an irregular 'graph' pyramid by successively contracting a dual image graph of one level into the smaller dual image graph of the next level. Dual image graphs are typically defined by the neighborhood relations of image pixels or by the adjacency relations of the region adjacency graph. The above concept has been used for finding the structure of connected components [17, 14]. It also embeds Meer's stochastic pyramid [21], the adaptive pyramid [9], and a further variant of Meer's approach, Mathieu's optimal stochastic pyramid [20] which produced excellent segmentation results by decimating a minimal spanning tree instead of the original graph.

The paper is organized as follows. We first summarize and illustrate the procedure of dual graph contraction in Section 2. The observation that the parameters that control the process form forests is then generalized by the concept of contraction kernels. Originally of depth one, deeper forests are now permitted and allow bigger contractions. They are necessary if repeated dual contractions are to be replaced by a single dual contraction using equivalent contraction kernels (section 2.2). ECKs are able to compute any level of an irregular pyramid directly from the base. Decimation parameters can be designed now at the base without the need to first generate the lower pyramid levels. The ECK of the apex becomes especially important in section 3. If labels are attached to the vertices and edges of this spanning tree all the individual decimation parameters can be recovered from this representation which is

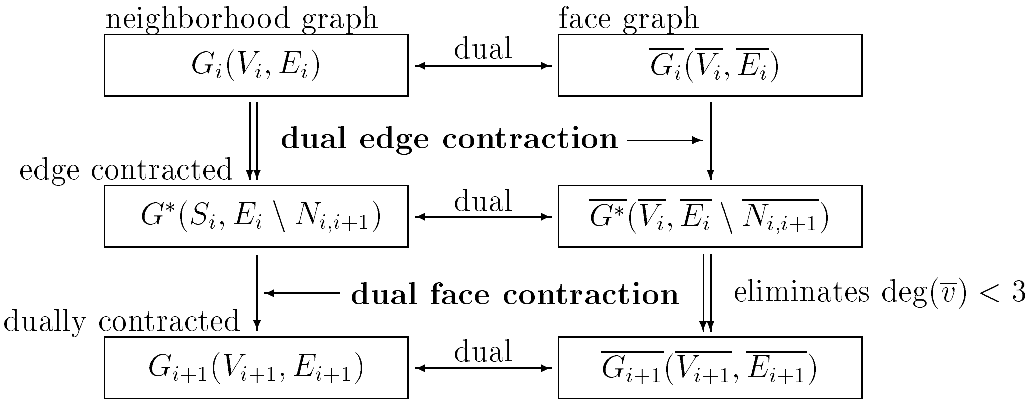


Figure 1: Dual Graph Contraction: $(G_{i+1}, \overline{G_{i+1}}) = C[(G_i, \overline{G_i}), (S_i, N_{i,i+1})]$

embedded in the base graph. As a consequence the labeled spanning tree (LST) determines the structure of the dual irregular pyramid completely. We therefore study in the sequel further methods to build (Section 4) and finally to modify this LST. It is the cue to define primitive operations (Section 5) that allow to explore the space of all possible dual irregular pyramids that can be built on top of a given base graph. The conclusion (Section 6) contains an outlook (Section 6.1) on potential further directions of research.

2 Dual Graph Contraction

Fig. 1 summarizes the two basic steps: dual edge contraction and dual face contraction. The base of the pyramid consists of the pair of dual image graphs $(G_0, \overline{G_0})$. We repeat the definition of the parameters determining the structure of an irregular pyramid given in [13][Def.5]:

Definition 1 *In a pair of dual image graphs* $(G_i(V_i, E_i), \overline{G_i}(\overline{V_i}, \overline{E_i}))$*, following* **decimation parameters** $(S_i, N_{i,i+1})$ *determine the contracted graphs* $(G_{i+1}, \overline{G_{i+1}})$*: a subset of* **surviving vertices** $S_i = V_{i+1} \subset V_i$*, and a subset of* **primary non-surviving edges**[1] $N_{i,i+1} \subset E_i$*. Every non-surviving vertex,* $v \in V_i \setminus S_i$*, must be connected to one surviving vertex in a unique way:*

$$\forall v \in V_i \setminus S_i \quad \exists s \in S_i : (v, s) \in N_{i,j}. \tag{1}$$

The relation between the two pairs of dual graphs, $(G_i, \overline{G_i})$ *and* $(G_{i+1}, \overline{G_{i+1}})$*, as established by dual graph contraction with decimation parameters* $(S_i, N_{i,i+1})$ *is expressed by function* $C[.,.]$*:*

$$(G_{i+1}, \overline{G_{i+1}}) = C[(G_i, \overline{G_i}), (S_i, N_{i,i+1})] \tag{2}$$

[1]Secondary non-surviving edges are removed during dual face contraction.

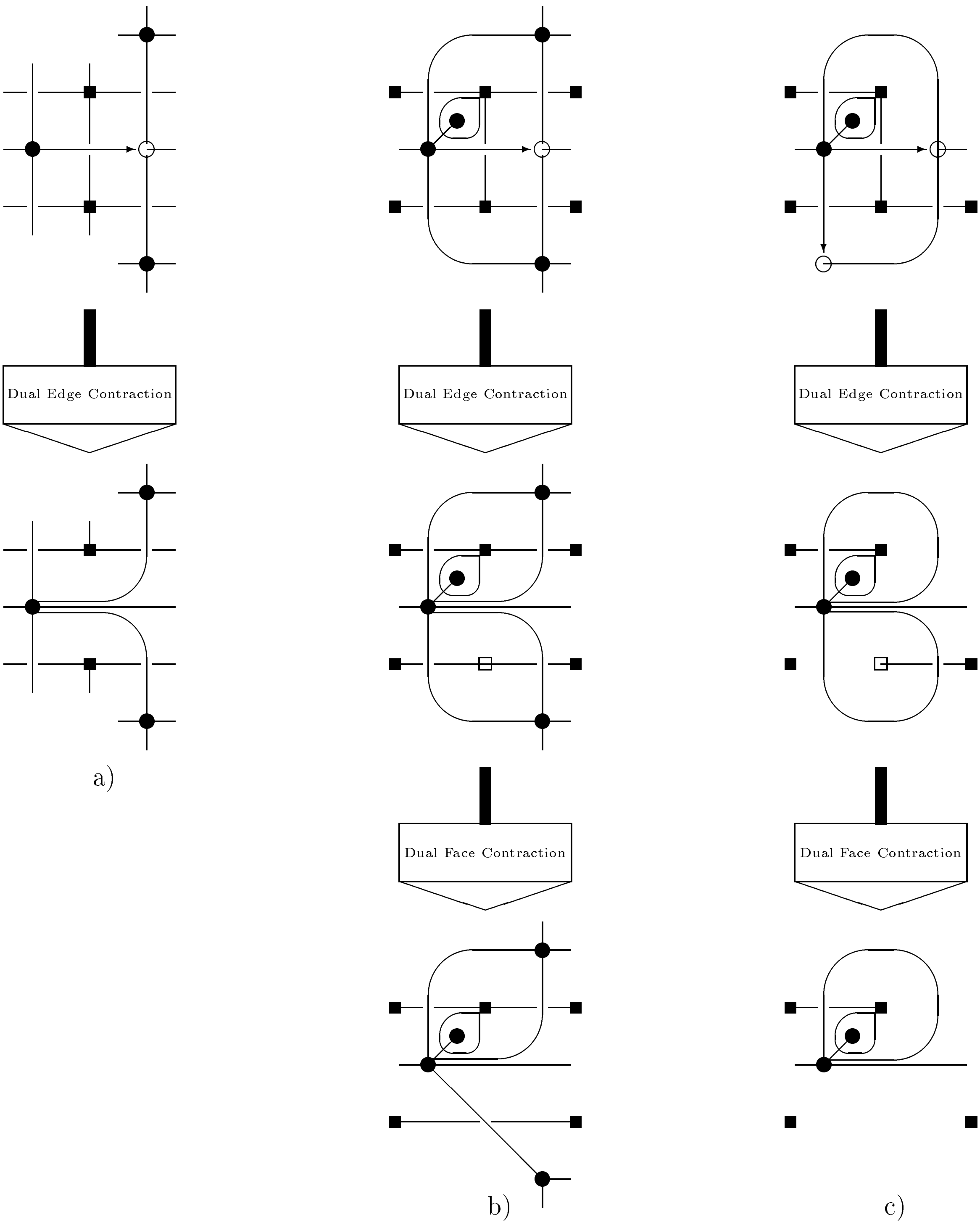


Figure 2: Three Cases of Dual Graph Contraction

Fig. 2 illustrates[2] the three different configurations around a primary non-surviving edge and explains also the different treatment. Fig. 2a) shows a normal dual edge contraction where no redundant faces are generated. The edges of the non-surviving vertex are drawn to its surviving parent. Multiple edges are generated in Fig. 2b) after dual edge contraction, and self-loops in Fig. 2c). The lower half of the figures shows a redundant face, while the upper halfs contain in both cases a surviving vertex. Dual face contraction can simplify the lower configurations, but not those in the upper half: any further contraction would change the existing neighbor relations.

Two steps of dual graph contraction shows the example of Fig. 3. They can be formally written as $(G_1, \overline{G_1}) = C[(G_0, \overline{G_0}), (S_0, N_{0,1})]$, and $(G_2, \overline{G_2}) = C[(G_1, \overline{G_1}), (S_1, N_{1,2})]$. Note that graph G_2 in this example contains both a self-loop and a double edge. It has been used in [13] to show the difference between three different types of graph contractions.

2.1 Decimation with Contraction kernels

Let us first reconsider the decimation parameters chosen in our example graph (Fig. 3d, e, f, levels $i = 0, 1, 2$ resp.). The connected components [3] $CC(s), s \in S$, of subgraph (S, N) form small tree structures $T(s)$ that collaps into vertex s of the contracted graph: $T(s) := (CC(s), N \cap (CC(s) \times CC(s)))$. Their union are the primary non-surviving edges N. $T(s)$ is a **spanning tree** of the connected component $CC(s)$, or equivalently, (V, N) is a spanning forest of graph $G(V, E)$. We therefore relax constraint (1) and require only the above observed properties:

Definition 2 *A decimation of a graph $G(V, E)$ is specified by a selection of* **surviving** *vertices $S \subset V$ and a selection of* **primary non-surviving edges** *$N \subset E$ such that following two conditions are fulfilled:*

1. *Graph (V, N) is a spanning forest of graph $G(V, E)$.*
2. *The surviving vertices $S \subset V$ are the roots of the forest (V, N).*

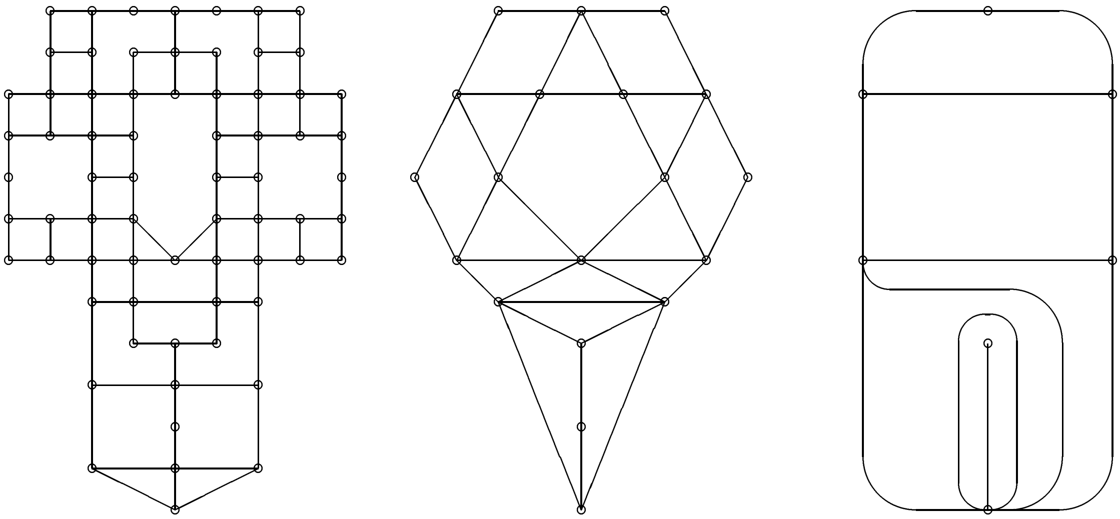
The trees $T(v)$ of the forest (V, N) with root $v \in V$ are called **contraction kernels**.

Instead of joining non-surviving vertices by an edge to their corresponding surviving parent vertex, the new concept establishes this connection via paths of non-surviving edges (e.g. branches of the trees). The concept of connecting path as introduced in [13][Def.6] is adapted accordingly:

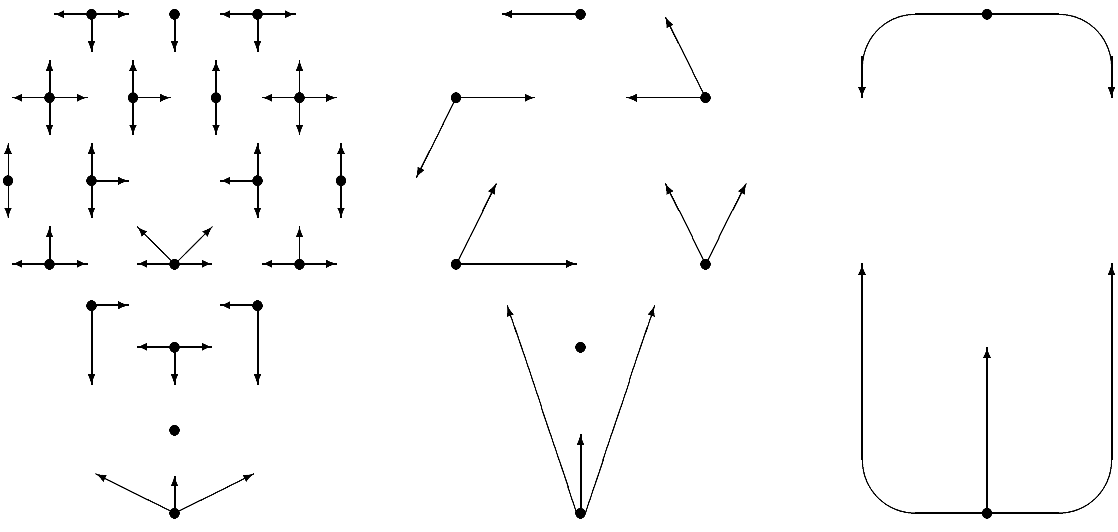
Definition 3 *Let G(V,E) be a graph with decimation parameters (S, N). A path in $G(V, E)$ is called a* **connecting path** *between two surviving vertices $v, w \in S$, denoted $CP(v, w)$, if it consists of three subsets of edges E (Fig. 4):*

[2]In figures, $S_i = \{\bullet\}$, $\overline{V_{i+1}} = \{\blacksquare\}$, $V_i \setminus S_i = \{\circ\}$, $\overline{V_i} \setminus \overline{V_{i+1}} = \{\square\}$ and $(\bullet, \circ) \in N_{i,j}$ are indicated by $\longrightarrow$.

[3]Neglected indices refer to contraction from level i to level $i+1$.



(a) (V_0, E_0) (b) (V_1, E_1) (c) (V_2, E_2)



(d) $(S_0, N_{0,1})$ (e) $(S_1, N_{1,2})$ (f) $(S_2, N_{2,3})$

Figure 3: Example of a dual irregular pyramid and decimation parameters

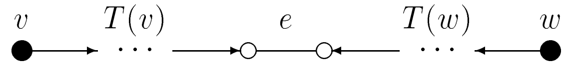
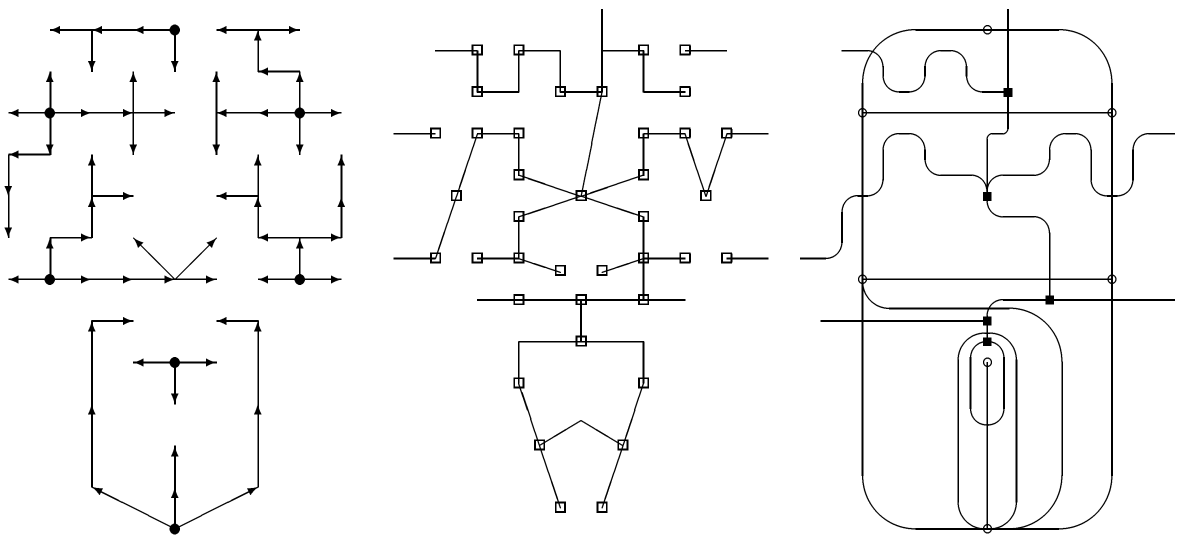


Figure 4: Decomposition of connecting path $CP(v, w)$

1. *The first part is a possibly empty branch of contraction kernel $T(v)$.*
2. *The middle part is an edge $e \in E \setminus N$ that bridges the gap between the two contraction kernels $T(v)$ and $T(w)$. We call e the* **bridge** *of the connecting path $CP(v, w)$.*
3. *The third part is a possibly empty branch of contraction kernel $T(w)$.*

Connecting paths $CP(v, w)$ in $G(V, E)$ are strongly related to the edges in the contracted graph $G'(V', E')$: Two different surviving vertices that are connected by a connecting path in G are connected by an edge in E'. For every edge $e' = (v, w) \in E'$ there exists a connecting path $CP(v, w)$ in G. Dual edge contraction can be implemented by (1) simply renaming all the non-surviving vertices to their surviving parent vertex, (2) deleting all non-surviving edges N and (3) their duals $\overline{N}$.



(a) $(V_2, N_{0,2})$ (b) $(\overline{V_0}, \overline{E_0} \setminus \overline{N_{0,2}})$ (c) $(V_2, E_2) \cup (\overline{V_2}, \overline{E_2})$

Figure 5: Example of equivalent contraction kernel

Fig. 5a shows different decimation parameters: Survivors $S = V_2$ are selected and the contraction kernels $N_{0,2}$ cover G_0. Like in a maze the edge-contracted face graph (Fig. 5b),

$\overline{G_0^*}(\overline{V_0}, \overline{E_0} \setminus \overline{N_{0,2}})$, fills in the holes left between the contraction kernels. Dual face contraction deletes all degree-one faces and shortens redundant connections established by the degree-two faces, resulting in $\overline{G_2}$. In Fig. 5c the preserved duality of $(G_2, \overline{G_2})$ can be verified.

2.2 Equivalent contraction kernels

Burt [6] introduced the 'equivalent weighting function': *"Iterative pyramid generation is equivalent to convolving the image g_0 with a set of 'equivalent weighting functions' h_l:"* $g_l = h_l * g_0 = h * g_{l-1}, l > 1$. It allowed him to study the effects of iterated reduction (e.g. the low-pass character of Gaussian pyramids) using the single parameter h_l without giving up the efficient iterative computation.

$$\begin{tikzcd}
\boxed{(G_{k-2}, \overline{G_{k-2}})} \arrow[r, "{(S_{k-2}, N_{k-2,k-1})}"] \arrow[rr, bend right, "{(S_{k-1}, N_{k-2,k})}"'] & \boxed{(G_{k-1}, \overline{G_{k-1}})} \arrow[r, "{(S_{k-1}, N_{k-1,k})}"] & \boxed{(G_k, \overline{G_k})}
\end{tikzcd}$$

Figure 6: Equivalent contraction kernel

Similarly we combine two (and more) dual graph contractions (see Fig. 6) of graph G_{k-2}, $k > 2$ with decimation parameters $(S_{k-2}, N_{k-2,k-1})$ and $(S_{k-1}, N_{k-1,k})$ into a single **equivalent contraction kernel** (ECK) $N_{k-2,k} = N_{k-2,k-1} \circ N_{k-1,k}$ (for simplicity G_i stands for $(G_i, \overline{G_i})$):

$$\begin{aligned} C[C[G_{k-2}, (S_{k-2}, N_{k-2,k-1})], (S_{k-1}, N_{k-1,k})] &= C[G_{k-2}, (S_{k-1}, N_{k-2,k})] \\ &= G_k \end{aligned} \tag{3}$$

Equivalent contraction kernels are constructed in the following way:

Assume that the dual irregular pyramid $((G_0, \overline{G_0}), (G_1, \overline{G_1}), \ldots, (G_k, \overline{G_k}))$, $k > 1$, is the result of k dual graph contractions. The structure of G_k is fully determined by the structure of G_{k-1} and the decimation parameters $(S_{k-1}, N_{k-1,k})$. Furthermore, the structure of G_{k-1} is determined by G_{k-2} and the decimation parameters $(S_{k-2}, N_{k-2,k-1})$. $S_{k-2} := V_k$ are the vertices surviving from G_{k-2} to G_k. The searched contraction kernels must be formed by edges $N_{k-2,k} \subset E_{k-2}$. This is true for $N_{k-2,k-1}$ but not for $N_{k-1,k} \subset E_{k-1}$ if we would simply overlay the two sets of decimation parameters. An edge $e_{k-1} = (v_{k-1}, w_{k-1}) \in N_{k-1,k}$ corresponds to a connecting path[4] $CP(v_{k-1}, w_{k-1})$ in G_{k-2}. By definition 3, $CP(v_{k-1}, w_{k-1})$ consists of one branch of $T_{k-2}(v_{k-1})$, one branch of $T_{k-2}(w_{k-1})$, and one surviving edge $e_{k-2} \in E_{k-2}$ connecting the two contraction kernels $T_{k-2}(v_{k-1}), T_{k-2}(w_{k-1})$.

[4]If there are more than one connecting paths, one must be selected.

Definition 4 *Function* **bridge**: $E_{k-1} \mapsto E_{k-2}$ *assigns to each edge* $e_{k-1} = (v_{k-1}, w_{k-1}) \in E_{k-1}$ *one of the bridges* $e_{k-2} \in E_{k-2}$ *of the connecting paths* $CP(v_{k-1}, w_{k-1})$*:*

$$bridge(e_{k-1}) := e_{k-2}. \tag{4}$$

Two disjoint tree structures connected by a single edge become a new tree structure. The result of connecting all contraction kernels T_{k-2} by bridges fulfills the requirements of a contraction kernel:

$$N_{k-2,k} := N_{k-2,k-1} \quad \cup \bigcup_{e_{k-1} \in N_{k-1,k}} \mathrm{bridge}(e_{k-1}) \tag{5}$$

The contraction kernels $(V_2, N_{0,2})$ in Fig. 5a are equivalent to the successive contraction with kernels of Fig. 3d and e.

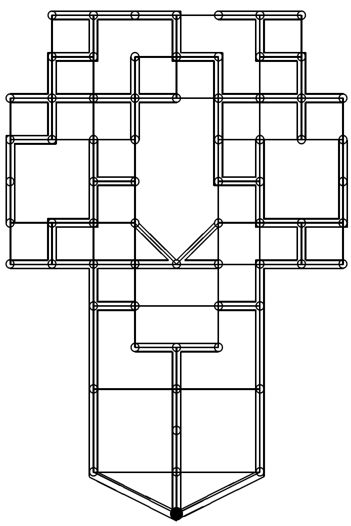


Figure 7: Example of ECK of apex: $G_0 \cup N_{0,4}$

The above process can be repeated on the remaining contraction kernels until the base level 0 contracts in one step into the apex $V_n = \{v_n\}$. The edges of the corresponding spanning tree are contained in $N_{0,n}$. Fig. 7 shows spanning tree $N_{0,4}$ overlaid with the base graph G_0. The apex, $v_4 \in V_4$, is marked by a filled circle and the edges of the spanning tree $N_{0,4}$ are differentiated from edges E_0 by triple lines.

2.3 Contracting the regions of a segmentation

One step to recover the structure of the scene from the projected structure in the image is to find the adjacency relations between the regions of a segmentation [17, 14]. Pyramids are computationally very efficient and can be used for segmentation. However they do not always produce satisfactory results (see [5]). Problems occure in particular if we require that every region of the segmentation is shrunk to only one pyramidal cell and that they are all represented at the same level of the pyramid. Then there exist counterexamples for all the regular pyramid segmentation algorithms as well as for all the irregular pyramids based on the maximum independent set. This is due to the fact that in all these pyramid schemes the factor of reduction is bounded, hence a segmentation with both a very small region and a very large region cannot appear at the same level. But also the size, the diameter, and the shape of the receptive fields cannot vary arbitrarily. In regular pyramids receptive fields have predefined regular shapes: squares, octagones, hexagones, ... (for more details see [12]).

Pyramid (re-)linking allows a fine tuning of region shapes by removing certain cells of the receptive field because they link to another ancester. But the resulting regions always fit inside the original region. Besides the restrictions imposed by the limited number of neighbors (due to regularity all interior cells have the same number of neighbors) the classical pyramid linking may also destroy the connectivity of the receptive fields [24]. Nacken's modifications not only preserve the connectivity of the receptive fields, but they also extend the original linking concept: links may move to any neighbor of a (newly) chosen parent even if it is not a neighbor of its original parent. As a consequence the number of neighbors of a cell may grow higher than in the initial state, and also the receptive fields can grow beyond the borders in the regular pyramid. Nacken's modification allows a much larger variety of shapes for receptive fields although it is not yet clear whether they span the same domain as in the dual irregular pyramids.

The new concept of contraction kernels allows different factors of contraction at different image regions. The following proposition proves that all possible segmentations (as defined in [25]) can be represented using contraction kernels. Note that any homogeneity predicate can be used to define the segmentation $\bigcup_{i=1}^{n} R_i$.

Proposition 1 *Let* $\bigcup_{i=1}^{n} R_i = V_0, R_i \cap R_j \neq \emptyset$ *be a partition of the vertex set into connected regions* R_i. *Then there exists a dual irregular pyramid* $((G_0, \overline{G_0}), (G_1, \overline{G_1}), \ldots, (G_k, \overline{G_k}))$ *built by dual graph contraction such that*

1. *All vertices* $v_k \in V_k$ *in the top level appear in exactly one region* R_i.
2. $card(V_k) = n$.
3. $card(R_i \cap V_k) = 1$ *for all regions* R_i.
4. *Let* $v_i \in R_i \cap V_k$ *and* $v_j \in R_j \cap V_k$, $i \neq j$; *then* $(v_i, v_j) \in E_k \Leftrightarrow R_i$ *and* R_j *are adjacent.*

Proof : (by construction of decimation parameters)

1. Cover each region R_i by a spanning tree $T_i \subset G_0$. This is possible since the regions are connected.
2. Select one vertex t_i as root in every spanning tree T_i.
3. Surviving vertices $S = \{t_i | i = 1, \dots, n\}$ and the edges N of all spanning trees T_i satisfy the conditions for decimation parameters, e.g. $\bigcup_{i=1}^{n} T_i$ forms a spanning forest of G_0.
4. Edges $e_k \in E_k$ of the dually contracted graph $(V_k, E_k) = C[G_0, (\{t_i\}, N)]$ are determined by connecting paths in G_0 with bridges $(v_i, v_j) \in E_0$ between region R_i and region R_j where $v_i \in R_i$ and $v_j \in R_j$.

This construction generates $(G_k, \overline{G_k})$ in a single step. We shall see in the following sections how the intermediate levels can be generated if necessary. For the proof the existence of such a pyramid is sufficient. ■

3 One representation for all selections

Bottom-up construction of a dual irregular pyramid $((G_0, \overline{G_0}), (G_1, \overline{G_1}), \dots, (G_n, \overline{G_n}))$ is formally described by dual contraction:

$$(G_{i+1}, \overline{G_{i+1}}) = C[(G_i, \overline{G_i}), (V_{i+1}, N_{i,i+1})] \qquad i = 0, 1, \dots, n-1 \tag{6}$$

$N_{i,i+1}$ denotes the subset of primary non-surviving edges at level i, e.g. $N_{i,i+1} \subset E_i$, that disappear during contraction of level i. Using the concept of equivalent contraction kernel,

$$N_{i,i+k} = N_{i,i+1} \circ N_{i+1,i+2} \circ \dots \circ N_{i+k-1,i+k} \qquad \forall 0 \le i, 0 \le k, i+k \le n \tag{7}$$

can be constructed such that any arbitrary level i can be directly derived from level 0:

$$(G_i, \overline{G_i}) = C[(G_0, \overline{G_0}), (S_{i-1}, N_{0,i})] \qquad i = 1, \dots, n \tag{8}$$

Following Table (9) summarizes all ECKs based on graph G_0. The left-side graph G_i can be dually contracted using contraction kernel $(V_k, N_{i,k})$ in the same row into graph G_k as shown in the last row. To produce higher pyramid levels with less vertices, obviously more edges need to be contracted as expressed by $N_{i,k} \subset N_{i,k+1}$.

$$\begin{array}{c|ccccc}
G_0 & N_{0,1} \subset & N_{0,2} \subset & N_{0,3} \subset & \dots \subset & N_{0,n} \\
G_1 & & N_{1,2} \subset & N_{1,3} \subset & \dots \subset & N_{1,n} \\
G_2 & & & N_{2,3} \subset & \dots \subset & N_{2,n} \\
\vdots & & & & \ddots & \vdots \\
G_{n-1} & & & & & N_{n-1,n} \\
\hline
 & G_1 & G_2 & G_3 & \dots & G_n
\end{array} \tag{9}$$

In a tree the number of vertices is one more than the number of edges. The decimation parameters $(V_{i+1}, N_{i,i+1})$ span the vertices V_i with $\mathrm{card}(V_{i+1})$ subtrees, hence

$$\mathrm{card}(V_i) = \mathrm{card}(N_{i,i+1}) + \mathrm{card}(V_{i+1}) \qquad \forall 0 \leq i < n \tag{10}$$

This equality must hold also for all ECKs:

$$\mathrm{card}(V_i) = \mathrm{card}(N_{i,k}) + \mathrm{card}(V_k) \qquad \forall 0 \leq i < k \leq n \tag{11}$$

From this we further derive the property

$$\mathrm{card}(N_{i,i+2}) = \mathrm{card}(N_{i,i+1}) + \mathrm{card}(N_{i+1,i+2}) \qquad \forall 0 \leq i < n-1 \tag{12}$$

and more generally

$$\mathrm{card}(N_{i,k}) = \sum_{j=i}^{k-1} \mathrm{card}(N_{j,j+1}) \qquad \forall 0 \leq i < k \leq n \tag{13}$$

Recalling that $N_{i,j}$ is a subset of ECK $N_{i,j+1}$ we can compute how many more edges must be contracted to get the next higher pyramid level:

$$\begin{aligned} \mathrm{card}(N_{i,j+1} \setminus N_{i,j}) &= \mathrm{card}(N_{i,j+1}) - \mathrm{card}(N_{i,j}) \\ &= \mathrm{card}(N_{j,j+1}) \qquad \forall 0 \leq i < j < n \end{aligned} \tag{14}$$

3.1 Labels indicate pyramid levels

Surviving vertices are ordered by set-inclusion, e.g. $V_0 \supset V_1 \supset \ldots \supset V_n$, as well as are the ECKs $N_{0,i}, i = 1, \ldots n$, in Table (9). Hence the vertex set V_0 and the edge set $N_{0,n}$ contain all decimation parameters needed to dually contract the base graph G_0 into any other pyramid level. We use labels to attach the whole construction history to the spanning tree $(V_0, N_{0,n})$ of the base graph G_0.

The vertices receive as a label the highest level to which they survive:

$$l(v) := k \Longleftrightarrow v \in V_k \setminus V_{k+1} \qquad \forall v \in V_0, 0 \leq k < n. \tag{15}$$

Edges receive the highest level as label to which they survive:

$$l(e) := k \Longleftrightarrow e \in N_{0,k+1} \setminus N_{0,k} \qquad \forall e \in N_{0,n}, 0 \leq k < n. \tag{16}$$

Figure 9a, b, show such a labeling for the levels 0, 1, 2, 3, and 4 in our example pyramid. The labels of the vertices in Fig. 9a are shown inside the circles identifying them. The bridges of $N_{0,4}$ are displayed as elongated rectangles surrounding the label. The edges of level 0 are indicated by straight lines and the vertices of level 0 are omitted to not overload the figure too much. Figure 8 shows an 'unfolded' tree of the same structure: The labeled vertices correspond to those in Fig. 9a, only the paths connecting the vertices have been 'unfolded'.

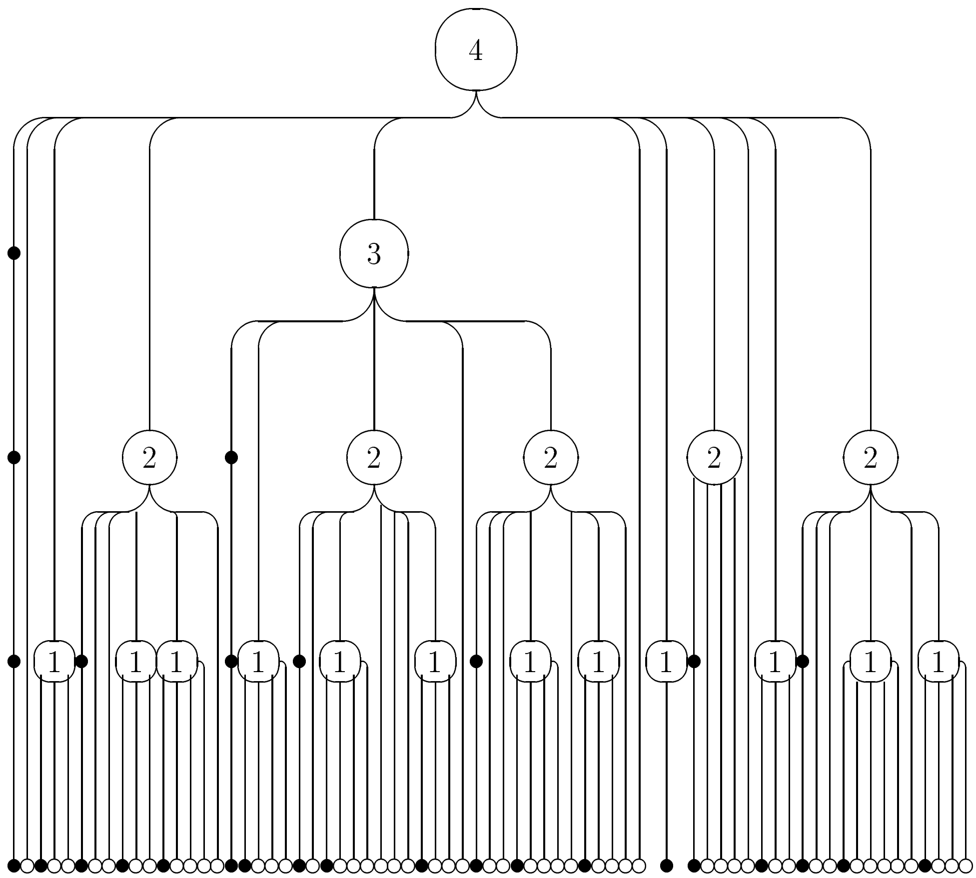
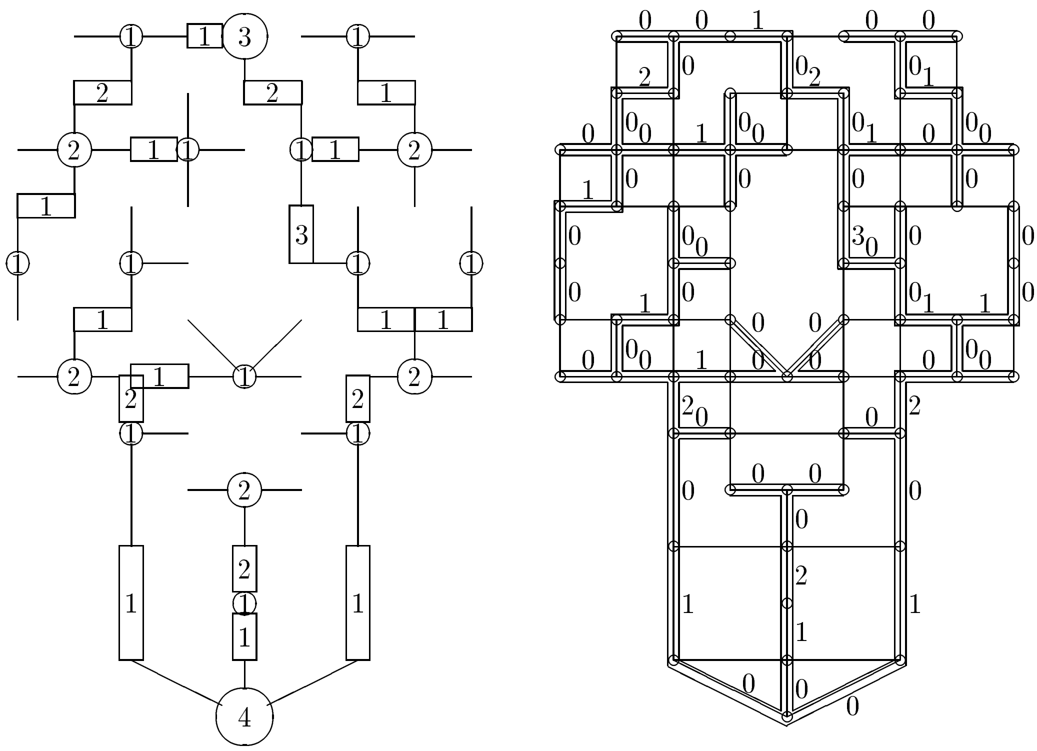


Figure 8: Unfolded tree of decimation parameters

Consider the path connecting the root vertex 4 with the vertex labeled 3. In Fig. 9a it collapses with other paths like to the vertex 2 located on this path which are distinguished in the unfolded representation (Fig. 8). The fact that higher level vertices also appear in the levels below is visualized by filled circles on the tree branches. It allows to count the vertices of any individual pyramid level.

The above labeling assigns labels to the tree $T = (V_0, N_{0,n})$ spanning the base graph $G_0 = (V_0, E_0)$. The following property of T extends this labeling to all edges E_0 of the base graph. The removal of any edge $t_0 = (u, w) \in N_{0,n}$ from the spanning tree T splits it into two (connected) components $V_0 = CC(u) \cup CC(w)$. One of them contains the apex, the other disappears after contracting level $k = l(t_0)$. By the definition of the edge labels $t_0 \in N_{0,k+1}$. If $k > 0$, there exists another edge $t_1 \in N_{1,k+1}$ such that $t_0 = \mathrm{bridge}(t_1)$ (Def. 4). In the same manner a sequence $(t_0, t_1, \ldots, t_k)$ can be found with $t_i = \mathrm{bridge}(t_{i+1})$ and $t_i \in N_{i,k+1}, 0 \leq i \leq k$. This sequence ends at $t_k \in N_{k,k+1}$ which is contracted at level $k + 1$.

If t_0 is not a bridge of graph G_0 then there exist other edges $e_0 \in E_0$ that connect $CC(u)$ and $CC(w)$, e.g. $e_0 \in CC(u) \times CC(w)$. Since e_0 has endpoints in different connected



(a) $l : V_0 \times N_{0,4} \mapsto [1, 4]$ (b) $l : N_{0,4} \mapsto [0, 3]$

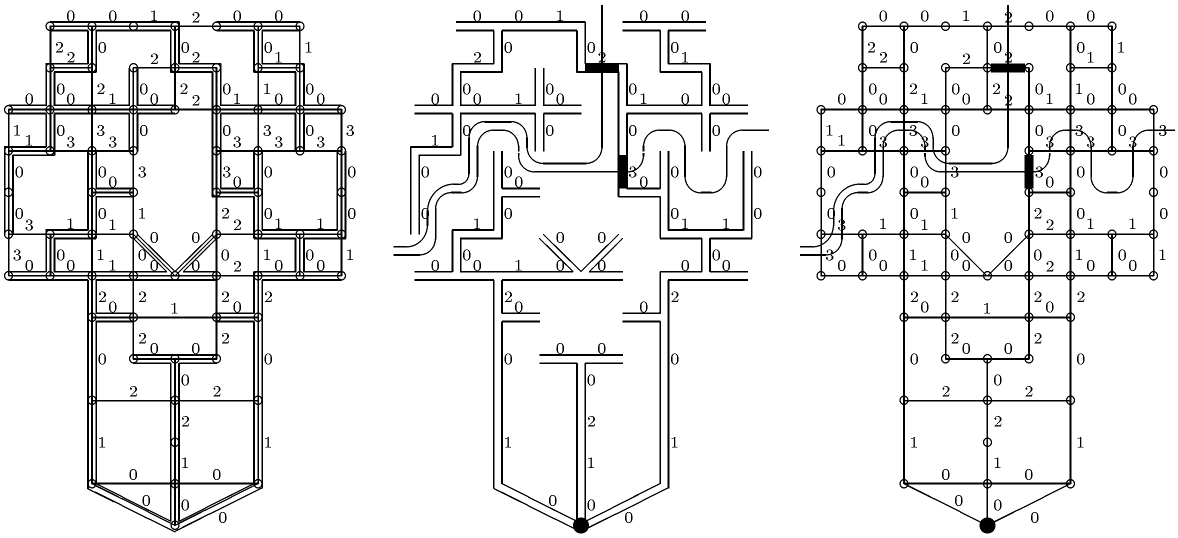
Figure 9: Labels of the spanning tree

components, a similar sequence $(e_0, e_1, \ldots, e_k)$ can be constructed with $e_i = \text{bridge}(e_{i+1})$ and $e_i \in E_i, 0 \leq i \leq k$. It can either merge with the sequence $(t_0, t_1, \ldots, t_k)$ at a level, e.g. $e_i = t_i, 0 < i \leq k$, or $e_k \neq t_k \in E_k$ survives up to level k. Since one of the connected components will disappear at level $k+1$ also e_k must be contracted. Hence e_0 receives the label $l(e_0) = k$ since k is the highest level to which e_0 survives.

Fig. 10b and c show two examples: Once the edge with the label 3 has been chosen, once t_0 carries label 2. They are highlighted in Fig. 10b, c by filled rectangles. The boundaries between the two induced connected components are drawn as continuous curves in both cases. In Fig. 10c the boundary curves are overlaid with the base graph and intersect a number of edges. Obviously these edges connect the same two connected components as the chosen t_0. It further shows that several edges of the base graph are intersected by both boundaries. In this case we choose the higher label:

$$l(e) := \max\{l(t_0) | \exists t_0 = (u, w) \in N_{0,n} \text{ such that } e \in CC(u) \times CC(w)\} \qquad \forall e \in E_0. \quad (17)$$

In this way all edges E_0 except self-loops receive a label (Fig. 10a).



(a) $l : E_0 \mapsto [0, 3]$ (b) 2 cuts of $l(N_{0,4})$ (c) 2 cuts of $l(E_0)$

Figure 10: Labels of the base graph

3.2 Reconstruction of decimation parameters

The hierarchy of surviving vertices can be reconstructed from V_0 by thresholding the vertex labels $l : V_0 \mapsto \{0, 1, \ldots, n\}$:

$$V_i = \{v \in V_0 | l(v) \leq i\} \qquad i = 1, \ldots, n \tag{18}$$

The ECKs for the base level, $N_{0,i}$, result similarly by thresholding the edge labels $l : N_{0,n} \mapsto \{0, 1, \ldots, n-1\}$:

$$N_{0,i} = \{e \in N_{0,n} | l(e) < i\} \qquad i = 1, \ldots, n-1 \tag{19}$$

The ECKs of the higher levels $j > 0$ can be derived by contracting lower level ECKs $k < j$:

$$(V_j, N_{j,i}) = C[(V_k, N_{k,i}), (V_j, N_{k,j})] \qquad \forall 0 \leq k < j < i \leq n \tag{20}$$

The two conditions that (20) is a valid decimation are quickly verified: (1) $(V_k, N_{k,j})$ is a forest spanning $(V_k, N_{k,i})$ since $N_{k,j} \subset N_{k,i}$ for $j < i$, and (2) V_j are the roots of $(V_k, N_{k,j})$ by construction. To see that the dual contraction $C[G_j, (V_i, N_{j,i})]$ with the reconstructed primary non-surviving edges from (20) reproduces the same G_i, we recall the inclusions $N_{k,j} \subset N_{k,i} \subset E_k$. We can assume that $N_{k,j}$ and $N_{k,i}$ are equivalent contraction kernels allowing to contract graph G_k into G_j and G_i respectively, e.g. $C[G_k, (V_j, N_{k,j})] = G_j$ and $C[G_k, (V_i, N_{k,i})] = G_i$. The result follows from considering connecting paths in G_k connecting vertices of G_i and the effects of contracting them with $N_{k,j}$.

Note that the contraction of the forest in (20) does not need dual face contraction because only the contraction of a cycle can generate redundant faces.

4 The domain of all irregular pyramids

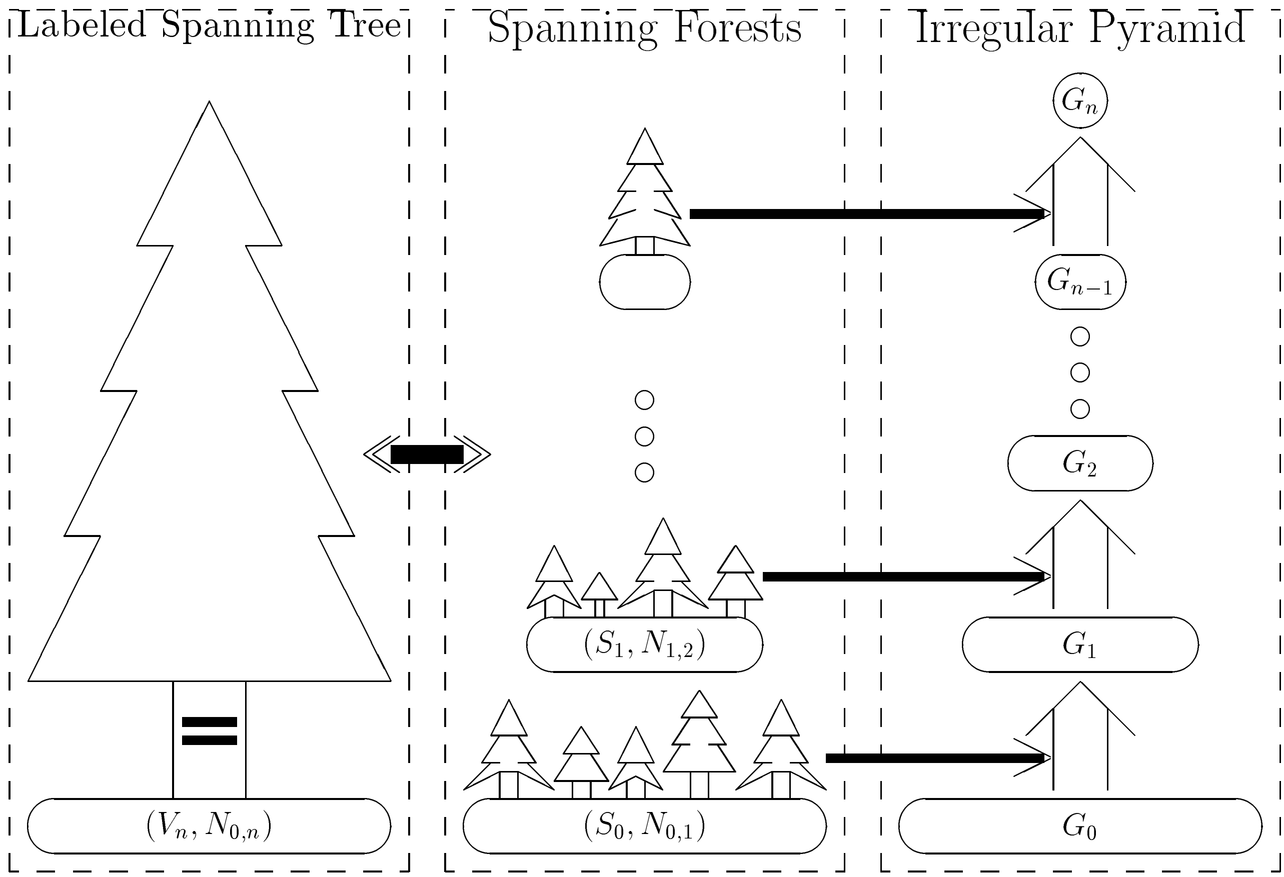


Figure 11: A labeled spanning tree combines all decimation parameters

Since decimation parameters $Dpar_i := (S_i, N_{i,i+1}), i = 0, \ldots, n-1$ determine the dual irregular pyramid $DPirr[(G_0, \overline{G_0}), Dpar_0, \ldots, Dpar_{n-1}] := ((G_0, \overline{G_0}), (G_1, \overline{G_1}), \ldots, (G_n, \overline{G_n}))$ completely and since all decimation parameters are equivalently represented by the labeled spanning tree $LST[V_0, N_{0,n}, l]$, every $DPirr$ is also determined by the labeled spanning tree LST, e.g. $DPirr[(G_0, \overline{G_0}), LST] = ((G_0, \overline{G_0}), (G_1, \overline{G_1}), \ldots, (G_n, \overline{G_n}))$ (Fig. 11). A systematic variation of the two components of the LST, e.g.

- the spanning tree $(V_0, N_{0,n})$ and
- the vertex and the edge labels $l(.)$

spans therefore the whole domain of dual irregular pyramids built on $(G_0, \overline{G_0})$. We shall first neglect the labels and study the domain of all spanning trees. Once we know the spanning tree we must define the labels such that the result corresponds to correct decimation parameters.

4.1 The domain of all spanning trees

Given a non-oriented, connected graph $G = (V, E)$. Denote by $T = (V, B)$ a spanning tree of G, e.g. $B \subset E$, $\text{card}(B) = \text{card}(V) - 1$, T does not contain any cycle, T is connected. An initial spanning tree T_0 can be computed by the following algorithm:

1. Select a root vertex v_0, set $V_T := \{v_0\}$ and $B := \emptyset$;
2. While $V \setminus V_T \neq \emptyset$ do steps 3 and 4;
3. find an edge $e = (v, w) \in E \setminus B$ with $v \in V \setminus V_T$ and $w \in V_T$;
4. $V_T := V_T \cup \{v\}$ and $B := B \cup \{e\}$.

Adding any further edge e_1 from $E \setminus B$ to T_0 necessarily creates a cycle. The removal of one edge e_2 from such a cycle regenerates another spanning tree. We define a transformation

$$modif[T, e_1, e_2] := (V, B \cup \{e_1\} \setminus \{e_2\}) \tag{21}$$

that modifies a spanning tree T_1 into another spanning tree T_2 of G. Furthermore we can show that this primitive operation allows to reach any arbitrary spanning tree in the domain of all spanning trees of G.

Proposition 2 *Let $T_1 = (V, B_1)$ and $T_2 = (V, B_2)$ be spanning trees of $G = (V, E)$. Then T_1 can be transformed into T_2 by a sequence of card$(B_2 \setminus B_1)$ primitive modifications (21).*

Constructive **Proof :** If $T_1 = T_2$ no modification is needed, e.g. $\text{card}(B_2 \setminus B_1) = 0$. Otherwise, we can select any edge $e_+ \in B_2 \setminus B_1$ which creates a cycle C in the graph $(V, B \cup \{e_+\})$. Since e_+ is part of T_2, which is a tree without cycle, not all edges of C can belong to T_2. Hence the edge e_- to be removed from C can be chosen in $B_1 \setminus B_2$ so we obviously get one more edge in common between T_2 and T_1. The process can be repeated until $B_1 = B_2$ which will be the case after any edge of $B_2 \setminus B_1$ has been integrated in T_1. ■

If we systematically add non-tree edges $e_+ \in E \setminus B$ and if we systematically remove edges $e_- \in C(e_+)$ of the generated cycles then $T := modif[T, e_+, e_-]$ will move through all possible spanning trees.

4.2 Labeling a spanning tree

Given the non-rooted spanning tree $(V_0, N_{0,n})$ the assignment of labels can proceed similar to the decimation process in a bottom-up process, but also top-down assignment is possible. The knowledge of the spanning tree allows several simplifications. At any given level k, an edge $e \in N_{k,n}$ is contracted if $l(e) = k, e \in N_{k,k+1}$, or it survives if $l(e) > k$, e.g. $e \in N_{k,n} \setminus$

$N_{k,k+1}$. Since $N_{k,k+1} \subset N_{k,n}$, $\mathrm{card}(N_{k,n} \setminus N_{k,k+1}) = \mathrm{card}(N_{k,n}) - \mathrm{card}(N_{k,k+1}) = \mathrm{card}(N_{k+1,n})$ using (13) for $N_{k,n}$ and $N_{k+1,n}$. Let $e_{k+1} \in N_{k+1,n}$, then $\mathrm{bridge}(e_{k+1}) \in N_{k,n} \setminus N_{k,k+1}$ are all distinct edges. Hence the property used in the following algorithms:

$$N_{k,n} = N_{k,k+1} \cup \mathrm{bridge}(N_{k+1,n}) \tag{22}$$

4.2.1 A bottom-up algorithm

We present a variant of the algorithm efficiently applied in [20]. Our version operates on the labels instead of performing repetitive contractions.

We start by initializing all labels to 0. Then we iteratively increment the labels based on the selection criteria. The following loop starts with level $k := 0$ and is repeated until $\mathrm{card}(V_k) = 1$:

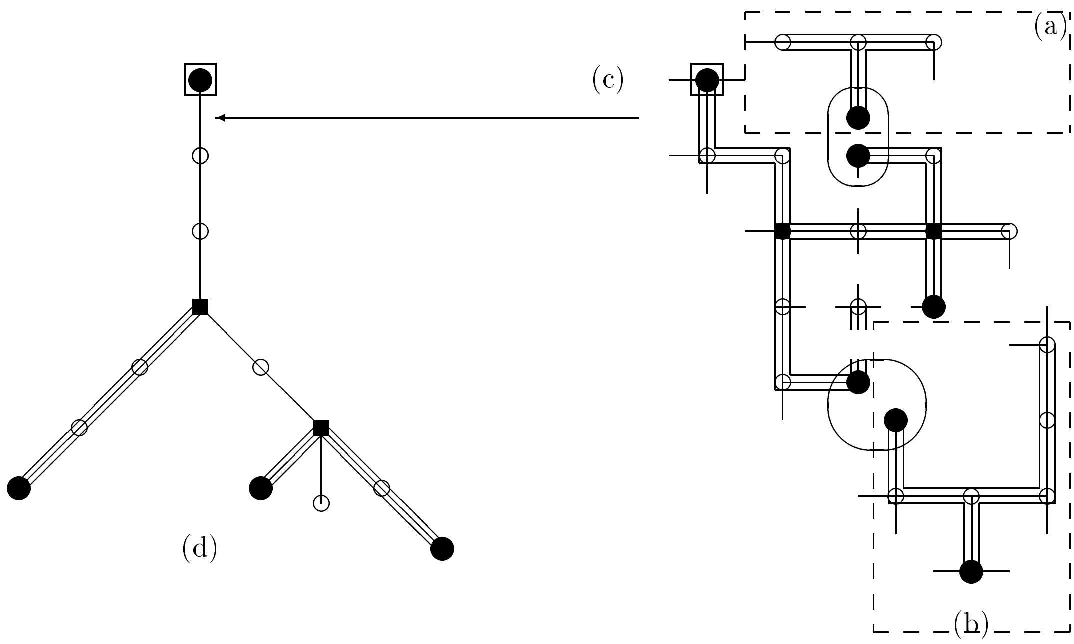


Figure 12: Three types of cuts

1. Select any survivors $V_{k+1} \subset V_k = \{v \in V_0 | l(v) = k\}$ and increment their labels.
2. The vertices of V_{k+1} cut the spanning tree into three types of connected subtrees (see Fig. 12a,b,c) which are processed differently:
   (a) There is only one survivor in the subtree. No edge must be incremented.

(b) The subtree contains exactly two survivors u and w: one edge with label k on the path connecting u and w in the subtree must be incremented.

(c) The subtree contains $m > 2$ survivors, only $m - 1$ edges are allowed to be incremented to yield a tree structure. One of the m survivors is selected to become a local root of the subtree. In the rooted subtree the other $m - 1$ survivors are leafs. The $m - 1$ edges with label k to be incremented must be located each on a different branch of the subtree connecting one of the leafs to its nearest branching point. (Illustrated in Fig. 12(d) by triple lines.)

3. $k := k + 1$

The selection of surviving vertices is the same as in decimation. But instead of growing the receptive fields of the survivors $N_{k,k+1}$ we select edges in $N_{k,n} \setminus N_{k,k+1}$ which will become the bridges for the next level.

Since bridges and primary non-surviving edges complement each other with respect to the contraction kernels according to (22) the selection of the bridges in the above algorithm implies automatically also the primary non-surviving edges. Hence the same decimation results can be achieved with the complementary selection.

4.2.2 A top-down algorithm

We start with the non-rooted spanning tree $(V_0, N_{0,n})$ and select any arbitrary vertex as the apex $v_n \in V_0, l(v_n) := n$.

The top-down construction proceeds recursively with label $k := n - 1$ by selecting and labeling the vertices and the edges not labeled in previous recursions. The recursion stops at the base level $k = 0$:

1. Select any subset of non-labeled vertices as V_k and assign them label k.

2. The labeled vertices $l(v) \geq k$ cut the spanning tree into three types of connected subtrees (similar to Fig. 12a,b,c) which are processed differently:

   (a) There is only one survivor in the subtree. No edge must be labeled.

   (b) The subtree contains exactly two labeled vertices u and w: one non-labeled edge on the path connecting u and w in the subtree receives label k.

   (c) The subtree contains $m > 2$ labeled vertices, only $m - 1$ edges need to be labeled to yield a tree structure. One of the m labeled vertices is selected to become a local root of the subtree. In the rooted subtree the other $m-1$ labeled vertices are leafs. The $m - 1$ non-labeled edges receiving label k must be located on branches (of the subtree) connecting those $m - 1$ leafs to their nearest branching point. (Illustrated in Fig. 12(d) by triple lines.)

3. Once all bridges of level k have been selected, we cut the tree at those edges into subtrees and repeat the process recursively with $k - 1$ until $k = 0$.

The selection of edges to be labeled must preserve connectivity as well as the tree structure. Hence only $m-1$ edges should be labeled in a tree connecting m labeled vertices.

The connection between any two subtrees T_1 and T_2 in step 2 is established by one labeled vertex that appears in both subtrees. Let m_1 be the number of labeled vertices in T_1 and m_2 in T_2. The total number of labeled vertices in both subtrees $T_1 \cup T_2$ is $m_1 + m_2 - 1$, the total number of labeled edges $(m_1 - 1) + (m_2 - 1)$. Hence $T_1 \cup T_2$ is again a connected tree.

A second split operation occurs in step 3 at edges labeled k. Assume that the two subtrees T_1 and T_2 are both connected and have m_1 and m_2 vertices and $m_1 - 1$ and $m_2 - 1$ edges respectively. The total number of vertices in both subtrees $T_1 \cup T_2$ is $m_1 + m_2$. If we include the cut-edge in the total number of edges, $(m_1 - 1) + (m_2 - 1) + 1$, $T_1 \cup T_2$ is again a connected tree.

5 Exploring the pyramid domain

A dual irregular pyramid is built on a pair of plane graphs $(G_0, \overline{G_0})$. Different decimation parameters yield different pyramidal structures on top of the same pair of plane graphs. We have described different ways to construct such a pyramid, we shall consider in this section primitive operations that transform one pyramid structure into another pyramid structure needing the least complex modifications, both concerning the decimation parameters and the graphs at the pyramidal levels.

5.1 Modify the labeled spanning tree

In section 4.1 (21) we have defined a transformation $T' = modif[T, e_+, e_-]$ that adds an edge e_+ to the spanning tree T and removes an edge e_- from T. The resulting graph T' is again a spanning tree.

In order to preserve the correct labeling of a labeled spanning tree the two edges must have the same label:

$$l(e_+) = l(e_-) \qquad e_+ \in E_0 \setminus N_{0,n}, \quad e_- \in N_{0,n} \tag{23}$$

The label of e_+ as computed in (17) makes sure that there is another edge e_- with the same label on the cycle introduced by the addition of e_+. Since the new edge e_+ connects the same connected components as e_- it represents only an alternative connecting path, and, consequently, the dual contraction produces the same result.

The primitive modification $modif[(V_0, N_{0,n}), e_+, e_-]$ with edges satisfying (23) changes the labeled spanning tree but not the corresponding irregular pyramid. It requires the modification of the labels in order to reach all the spanning trees of the unconstrained version.

5.2 Increment and decrement labels

The second primitive modification concerns the labels of the spanning tree. Besides the motivation to allow new structural changes through *modif*, it may also turn out that some elements in the pyramid are more important and should survive to higher levels whereas others could be contracted at lower levels.

Consider the contraction from level i to level $i+1$, the vertices to be spanned by the decimation parameters remain the same, e.g. V_i. In (10) we have established a relation between the number of vertices at two adjacent pyramid levels and the primary non-surviving edges at a given level i. Consequently, an additional surviving vertex in V_{i+1} requires one less edge (in $N_{i,i+1}$) to be contracted. Or conversely, every additional edge in $N_{i,i+1}$ must reduce the number of survivors by one. Hence labels of vertices and edges must be changed in parallel.

Another property to be respected by any modification is that all contraction kernels $(V_i, N_{i,n})$ must remain connected at higher levels. We should therefore increment only those edges in $N_{i,i+1}$ that are adjacent to $(V_{i+1}, N_{i+1,n})$, and conversely, decrement only the leafs of $(V_{i+1}, N_{i+1,n})$.

Definition 5 *Let $e = (v, w) \in N_{i,n}, l(e) = i$ be the edge to be incremented from level i to level $i+1$. One of the endpoints must have the same label, i.e. $l(v) = i$. Since this vertex cannot be adjacent to $(V_{i+1}, N_{i+1,n})$, the other vertex w must provide this property: $l(w) > i$. In this case the labels of both e and v can be incremented: $Incr[LST, e, v]$.*

Connectivity preservation excludes edges $e = (v, w)$ with $l(v) = l(w)$ from being incremented. However there is always a (unique) path towards the apex on the spanning tree $(V_i, N_{i,n})$ connecting such an edge to one that can be incremented. The edges of this path can be incremented backwards step by step, or, all edges and vertices of this path can be incremented at once.

If elements directly below the apex have been raised in level a new apex must be selected among the vertices V_n. This adds a new level to the pyramid.

For decrementation we have a similar constraint.

Definition 6 *Let $e = (v, w) \in N_{i+1,n}, l(e) = i+1 > 0$ be the edge to be decremented from level $i+1$ to level i . One of the endpoints must be a leaf of $(V_{i+1}, N_{i+1,n})$, i.e. $\deg(v) = 1$. Then the labels of both e and v can be decremented: $Decr[LST, e, v]$.*

Connectivity preservation excludes non-leaf vertices $\deg(v) > 1$ in $(V_{i+1}, N_{i+1,n})$ from being decremented. However v is the root of a (unique) subtree of $(V_{i+1}, N_{i+1,n})$ not containing the apex which can be decremented successively starting at its leafs. Or, all edges and vertices of this subtree can be decremented at once. The decrementation of the apex requires the selection of a new apex.

Operations $Incr$ and $Decr$ span the whole domain since any labeling can be decremented successively to the base level 0, and re-incremented to any other legal labeling after selection of the new apex.

Incrementation of level i and decrementation of level $i+1$ affect only the pyramid level $i+1$, hence requiring corrections in the irregular pyramid only at level $i+1$: $(G'_{i+1}, \overline{G'_{i+1}}) = C[(G_i, \overline{G_i}), (V'_{i+1}, N'_{i,i+1})]$ corrects the graphs for the modified labels. Although computational complexity of parallel dual contraction of a graph is low [28], one decrementation of the LST needs only one dual contraction of a single edge to adapt the pyramid: $Decr[LST, e, v]$ makes v a non-surviving leaf which is connected to the contraction kernel by e. If the other endpoint w of $e = (v, w)$ survives, e.g. $w \in V_{i+1}$, then dually contracting e gives the correct result: $(G'_{i+1}, \overline{G'_{i+1}}) = C[(G_{i+1}, \overline{G_{i+1}}), (V_{i+1} \setminus \{v\}, \{e\})]$.

5.3 Skip a pyramid level

An attractive alternative to reduce the height of the pyramid is to skip a whole level k using the ECKs: $skip\_level[((G_0, \overline{G_0}), (G_1, \overline{G_1}), \ldots, (G_n, \overline{G_n})), k]$. Besides deletion of graphs $(G_k, \overline{G_k})$ it consists in the deletion of the column $N_{i,k}, 0 \leq i < k$ and of the row $N_{k,j}, k < j \leq n$ from Table (9). Equivalently, all labels greater than k in the labeled spanning tree can be decremented by one. In the example of Fig. 13 levels 1 and 3 are skipped. In addition to our drawing conventions we have marked the six bridges $N_{0,4} \setminus N_{0,2}$ of level 2.

5.4 Insert an additional level

Decimation parameters $(S_k = V_k, N = \emptyset)$ duplicate a level of the pyramid. Such a contraction kernel can be used to insert a (dummy) level in the pyramid e.g. before labels of the higher level are decremented: $insert\_level[((G_0, \overline{G_0}), (G_1, \overline{G_1}), \ldots, (G_n, \overline{G_n})), k] = ((G_0, \overline{G_0}), \ldots, (G_k, \overline{G_k}), (G_k, \overline{G_k}), \ldots, (G_n, \overline{G_n})$. All labels of the LST that are not less than k must be incremented. The operation on the pyramid consists in an insertion of another copy of $(G_k, \overline{G_k})$. Insertion can be useful, if incrementation takes longer than decrementation, in particular when the graphs $(G_k, \overline{G_k})$ need to be adapted.

5.5 Summary of primitive modifications

We have seen that an edge e_- cannot be removed from the cycle created by addition of e_+ to the LST if $l(e_-) \neq l(e_+)$. By either incrementing the edge with the smaller label or decrementing the edge with the higher label the difference can be reduced stepwise until both edges have the same label.

We conclude this section by summarizing the discussed primitive modifications. The following table lists for all the operations their application test, and the computational complexity both to update the labeled spanning tree (LST) and to update the structure of the dual irregular pyramid (DPirr).

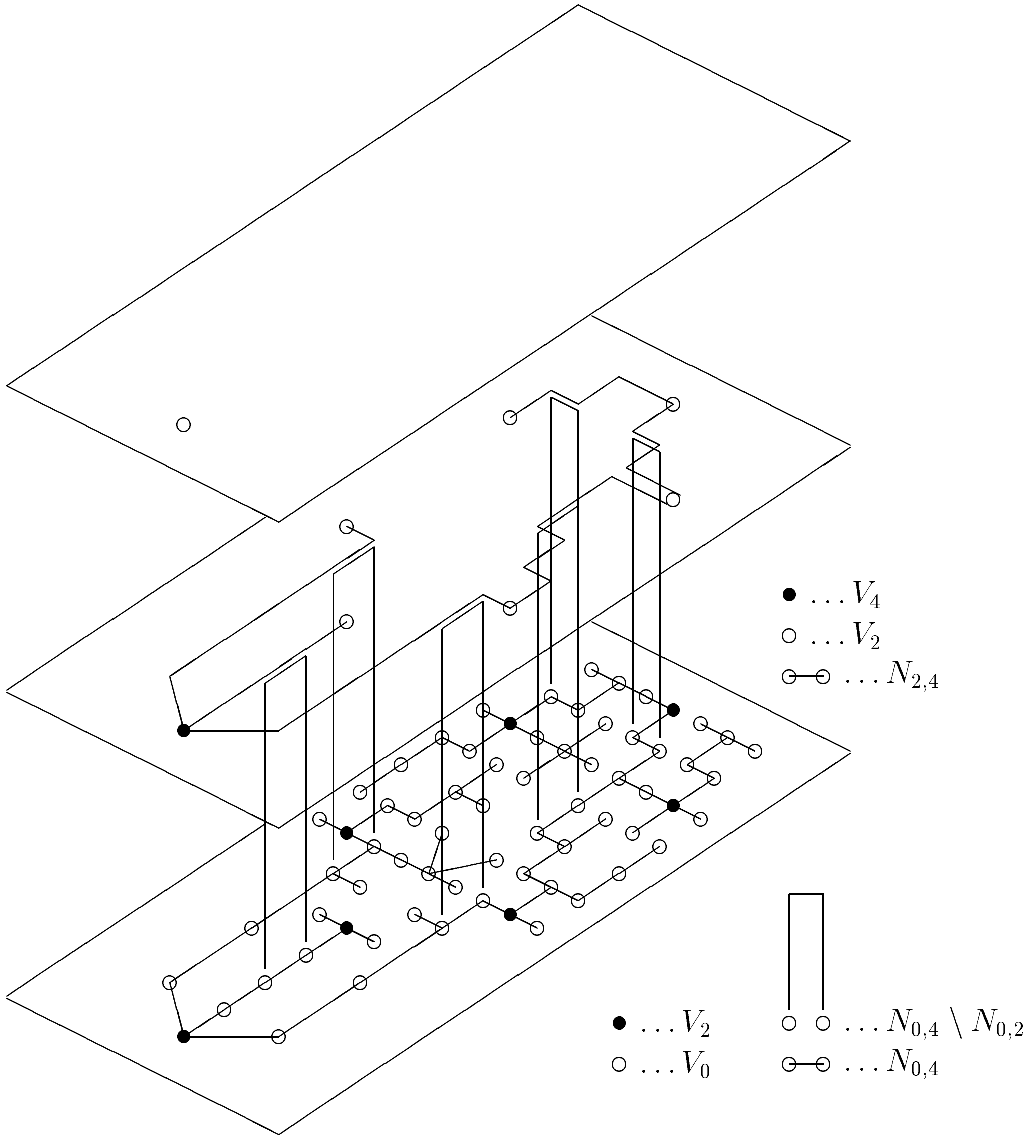


Figure 13: Decimation parameters after skipping levels 1 and 3

Operation	Test	$\mathcal{O}(LST)$	$\mathcal{O}(DPirr)$
$modif[(V_0, N_{o,n}), e_+, e_-]$	$l(e_+) = l(e_-)$	$\mathrm{card}(cycle(e_+))$	0
$Incr[LST, e = (v, w), v]$	$l(e) = l(v) < l(w)$	2	$\mathcal{O}(C[.,.])$
$Decr[LST, e = (v, w), v]$	$v = \mathrm{leaf}(V_{l(e)}, N_{l(e),n})$	2	$\mathcal{O}(C[., 1edge])$
$skip\_level[DPirr, k]$	none	$\mathrm{card}(V_{k+1}) + \mathrm{card}(E_{k+1})$	1
$insert\_level[DPirr, k]$	none	$\mathrm{card}(V_k) + \mathrm{card}(E_k)$	$\mathcal{O}(\mathrm{copy})$

6 Conclusion

Decimation parameters control dual graph contraction, a process that iteratively builds an irregular (graph) pyramid. The new concept of contraction kernel preserves the graph's structural properties, its connectivity, its planarity, and the face degrees of its dual graph.

Equivalent contraction kernels (ECKs) allow to skip the construction of intermediate pyramid levels. The contents of aggregations of cells can be computed efficiently and in parallel through the tree structure of the contraction kernels. The ECK of the apex is a spanning tree of the base graph. By attaching labels to the vertices and to the edges of this spanning tree we pack the decimation parameters of all pyramid levels into one single equivalent structure, the labeled spanning tree (LST), which is additionally a substructure of the base graph.

These new (global) decimation parameters can be computed without dually contracting the graphs at every step. Instead we can first generate a spanning tree and then determine the labels. We presented both a bottom-up and a top-down algorithm.

Once a labeled spanning tree and the corresponding dual irregular pyramid have been created it may turn out that certain modification are still desirable. Primitive operations $modif$, $Incr$, $Decr$, $skip\_level$, and $insert\_level$ allow the direct manipulation of the pyramid without complete reconstruction. This opens the possibilities to optimize the pyramidal structure in the pyramid domain and to dynamically adapt the structure to a changing input.

6.1 Outlook

The presented framework opens a large variety of issues for further investigation. The following list covers a large spectrum of ideas but makes no claim to be complete:

1. Structural issues:

    - **in 2D:**

        - Overlapping regular pyramids have several useful properties like robustness [22]. Our contraction kernels generate only non-overlapping receptive fields, hence the question how to combine **overlap** with dual graph contraction.
        - Computer architectures often bound the number of links of a processing element. Therefore the one-to-one mapping between irregular pyramid and architecture is not always possible. To overcome such **technological constraints** following issues could be studied: (1) integrate the constraints into the construction and modification algorithms; (2) simulate the full pyramid domain on a given architecture; (3) approximate the pyramid that cannot be mapped onto the architecture by the pyramid that is 'closest' in the pyramid domain and that allows such a mapping.

  - Represent uncertainty in spatial localization of boundary segments: **boundary regions** with curve relations representing the uncertain boundary segments (as in [15]).
  - Measure the computational effort to transform one LST into another LST and the corresponding pyramids in the overall domain. Tanaka's **tree metric** could be used [27].
  - Given the primitive operations in the pyramid domain, general **multi-level optimization** (with genetic algorithms?) could be applied, but what objective function should be used?
  - **Multi-pyramids:** Let the elements of the base graph have multi-valued attributes, e.g. a feature vector. Every such attribute can be used to build a (in principle different) pyramid that best represents the spatial distribution of the chosen attribute. It results a vector of pyramids with the same base. How can these pyramids be combined, either by an enhanced construction or by application of primitive operations in the pyramid domain?

- **in 3D and higher:**
  The advantages of pyramids should certainly not be restricted to two dimensional data. The third spatial dimension as well as the time dimension offer several interesting applications. In these cases the concepts of planarity and duality must be substituted by a higher dimensional concept that allows decisions whether a cell or a collection of cells is **inside** any other connected agglomeration of cells.
  - Four elements can be used to build structures in 3D: **pointels, linels, surfels, and voxels** (see i.e. [2]). These basic elements describe 0-, 1-, 2-, and

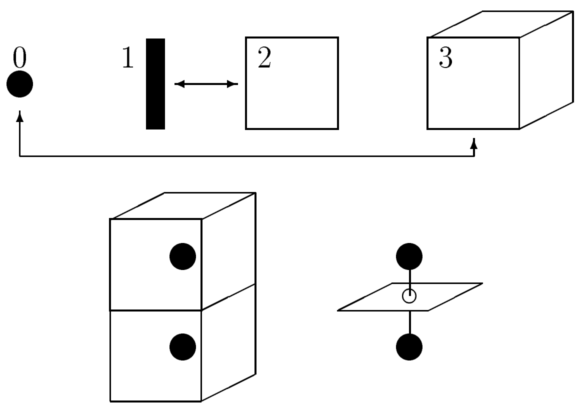


Figure 14: Duality in 3D: pointels, linels, surfels, and voxels

3-dimensional entities in 3D-space. Duality can be established by placing one pointel inside any voxel and by intersecting any surfel by one linel (Fig. 14).

– Voronoi diagram of a set of points defines a tesselation graph, and in 2D also its dual graph, the Delaunay triangulation. Voronoi diagram is defined also in higher dimension, could we use the induced graphs and duals?
– **Abstract cellular complexes** of Kovalevsky [10] offer another possibility to reach higher dimensions. Unfortunately this theory does not yet provide

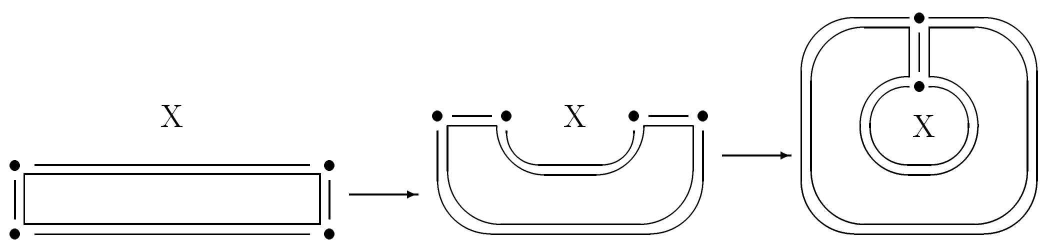


Figure 15: Bending a cell around a hole-region X

the possibility to represent holes. One idea to integrate holes is to bend a regular cell around the hole-region (marked 'X' in Fig. 15).
– The concepts of **star topology** [2] and **boundary graphs** of Ahronovitz [3] may be suited to be combined with hierarchical structures. The problem to be solved seems to be the convexity of cells which can be lost during contraction.

2. How to treat the content of a cell:

   - **Selection criteria:**
     – graph theoretic clustering [19];
     – stochastic LST, a combination of [21] and [20];
     – Integrate with selection and recovery paradigms [16];
     – interactive selection of decimation parameters.
   - **Complex models:**
     – parametric models as used in ExSel++ [26];
     – symbolic and fuzzy curve relations [4];
     – region-contour interaction, in continuation of [11, 23, 15];
     – Combine and use robust statistics of Förstner to derive his feature adjacency graphs (FAG) [7, 8] which match well with the graphs in our approach.

3. Applications:

   - **Segmentation** with the presented approach needs specification of selection criteria, of models describing the content of a pyramidal cell, of reduction and refinement functions that propagate information up and down in the pyramid, and of criteria for global adaptation in the pyramid domain.

- **Structural deblur:** The approach presented in [14] finds the extended region adjacency graph for ideal data. Blurring a gray level image normally introduces intermediate values along the boundaries between regions, giving rise to new regions and more complicated adjacency graphs. The aim of structural deblur would be to derive the adjacency graph of ideal data from its blurred version by primitive operations in the pyramid domain.
- **Dynamic adaptation:** Dynamic adaptation should be naturally linked to dynamic changes in an images. Hence approaches like optic flow, motion, etc. should be investigated.
- **Find correspondences between images** for stereo and for motion.
- **Model-guided interpretation:** The knowledge of an object composition determines both (1) what spatial configuration the object's individual parts (each represented in one pyramidal cell) must have and how they can be agglomerated into their parent cell (bottom-up abstraction); and (2) where and what type of object part should be searched for if the presence of the object is hypothesized; such substitution grammars could parse the pyramid domain of given image data. Fig. 16 illustrates the concept by a simple example of a bug ($b \in V_{k+1}$) consisting

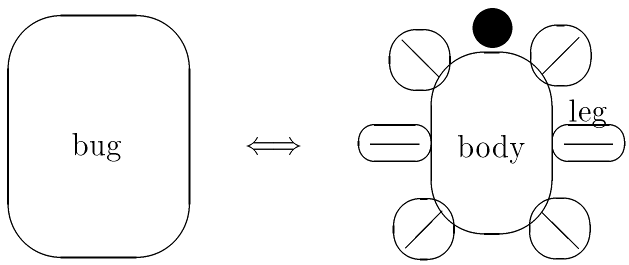
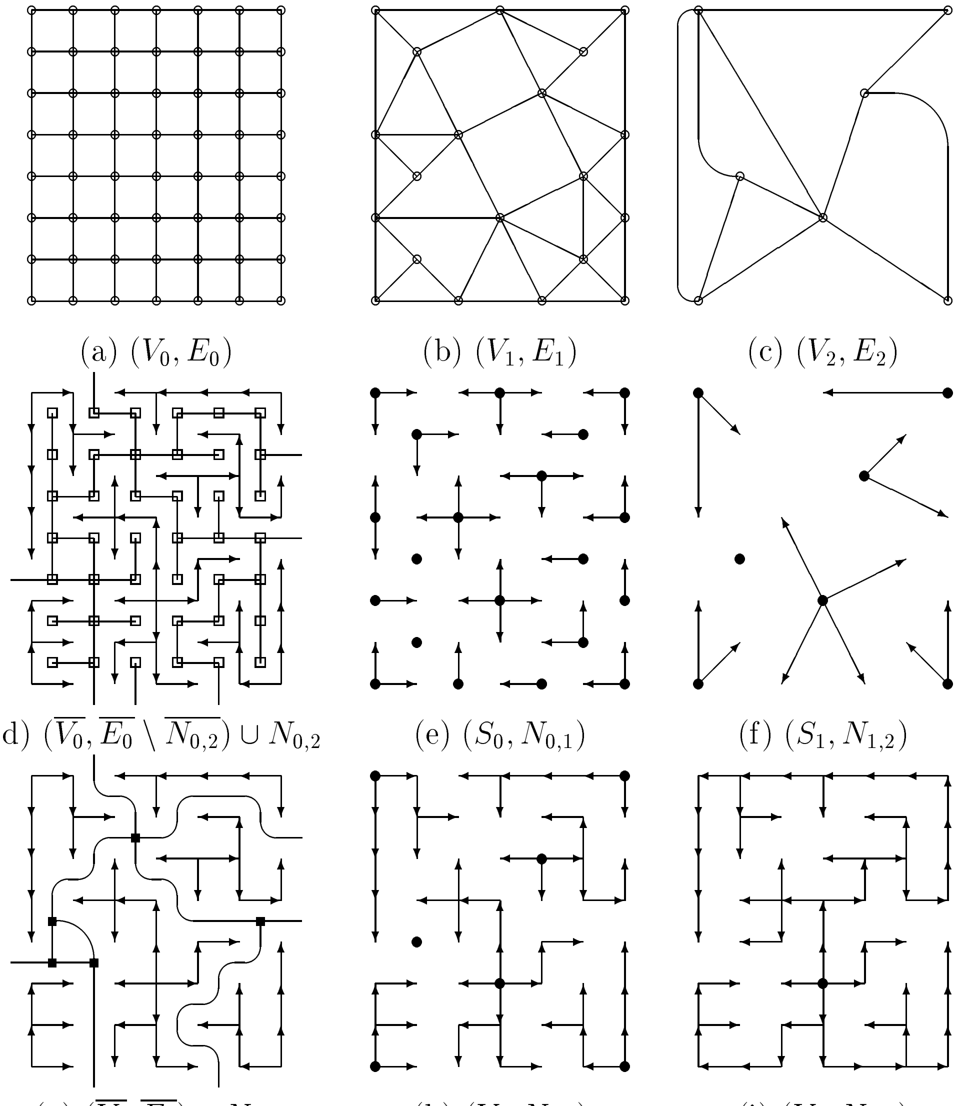


Figure 16: Decomposition of a bug: body, head, and 6 legs.

of a body $b \in V_k$, a head $h \in V_k$, and six legs $l_1, \ldots l_6 \in V_k$.

Acknowledgement

The author would like to thank Jean-Michel Jolion and Annick Montanvert for several encouraging discussions. LIP-ENS Lyon provided the environment during the author's sabbatical where a major part of this work was done.



(a) (V_0, E_0) (b) (V_1, E_1) (c) (V_2, E_2)

(d) $(\overline{V_0}, \overline{E_0} \setminus \overline{N_{0,2}}) \cup N_{0,2}$ (e) $(S_0, N_{0,1})$ (f) $(S_1, N_{1,2})$

(g) $(\overline{V_2}, \overline{E_2}) \cup N_{0,2}$ (h) $(V_2, N_{0,2})$ (i) $(V_n, N_{0,n})$

Figure 17: Example of a dual irregular pyramid and decimation parameters

Appendix A

Figure 17 presents another example of a dual irregular pyramid which is constructed on top of a regular grid. Figures 17(a), (b), (c) show levels 0, 1, 2 obtained with the decimation parameters in Fig. 17(e) and (f) respectively. Figures 17(d) and (g) show the dual graphs before and after dual face contraction with the ECK shown in Fig. 17(h). Fig. 17(i) is the ECK of the apex.

References

[1] Peter K. Ahnelt and Dietmar Pum. The foveal photoreceptors. a model for the comparison of physiologic and algorithmic image compressions. In Peter Mandl, editor, *Modelling and New Methods in Image Processing and in Geographical Information Systems*, OCG-Schriftenreihe, Österr. Arbeitsgemeinschaft für Mustererkennung, pages 223–232. Oldenbourg, 1991. Band 61.

[2] Ehoud Ahronovitz, Jean-Pierre Aubert, and Christophe Fiorio. The star-topology: a topology for image analysis. In *Cinquième Colloque DGCI*, pages 107–116. LLAIC1, Université d'Auvergne, ISBN 2-87663-040-0, September 1995.

[3] Ehoud Ahronovitz and Christophe Fiorio. Un système topologique complet pour le traitement d'images. Technical Report Research Report 94-044, Lab. d'Informatique, de Robotique et de Microélectronique de Montpellier, December 1994.

[4] Horst Bischof and Walter G. Kropatsch. The fuzzy curve pyramid. In Shmuel Peleg, Shimon Ullman, and Yehezkel Yeshurun, editors, *12th IAPR International Conference on Pattern Recognition*, volume I, pages 505–509. IEEE Comp.Soc., 1994.

[5] M. Bister, J. Cornelis, and Azriel Rosenfeld. A critical view of pyramid segmentation algorithms. *Pattern Recognition Letters*, Vol. 11(No. 9):pp. 605–617, September 1990.

[6] P. J. Burt and E. H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, Vol. COM-31(No.4):pp.532–540, April 1983.

[7] Wolfgang Förstner. A Framework for Low Level Feature Extraction. In Jan-Olof Eklundh, editor, *Computer Vision - ECCV 94, Vol. II*, pages 383–394, 1994.

[8] Claudia Fuchs and Wolfgang Förstner. Polymorphic Grouping for Image Segmentation. In Eric Grimsen, editor, *5th ICCV 95*, pages 175–182, 1995.

[9] Jean-Michel Jolion and Annick Montanvert. The adaptive pyramid, a framework for 2D image analysis. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 55(3):pp.339–348, May 1992.

[10] Vladimir A. Kovalevsky. Digital Geometry Based on the Topology of Abstract Cellular Complexes. In Jean-Marc Chassery, Jean Francon, Annick Montanvert, and Jean-Pierre Réveillès, editors, *Géometrie Discrète en Imagery, Fondements et Applications*, pages 259–284, Strasbourg, France, September 1993.

[11] Walter G. Kropatsch. Preserving contours in dual pyramids. In *Proc. 9th International Conference on Pattern Recognition*, pages 563–565, Rome, Italy, November 1988. IEEE Comp.Soc.

[12] Walter G. Kropatsch. Rezeptive Felder in Bildpyramiden. In H. Bunke, Olaf Kübler, and P. Stucki, editors, *Mustererkennung 1988*, Informatik Fachberichte 180, pages 333–339. Springer Verlag, 1988.

[13] Walter G. Kropatsch. Building Irregular Pyramids by Dual Graph Contraction. Technical Report PRIP-TR-35, Institute f. Automation 183/2, Dept. for Pattern Recognition and Image Processing, TU Wien, Austria, 1994.

[14] Walter G. Kropatsch and Herwig Macho. Finding the structure of connected components using dual irregular pyramids. In *Cinquième Colloque DGCI*, pages 147–158. LLAIC1, Université d'Auvergne, ISBN 2-87663-040-0, September 1995.

[15] Walter G. Kropatsch and Dieter Willersinn. Parallel line grouping in irregular curve pyramids. In *Proceedings Computer Vision and Pattern Recognition - CVPR'93*, pages 784–785. IEEE Comp.Soc.Press, 1993.

[16] Aleš Leonardis. Robust Recovery of Structures in Images. *Computing Supplementum: 'Theoretical Foundations of Computer Vision'*, pages –, 1996. In press.

[17] Herwig Macho and Walter G. Kropatsch. Finding Connected Components with Dual Irregular Pyramids. In Franc Solina and Walter G. Kropatsch, editors, *Visual Modules, Proc. of 19th ÖAGM and 1st SDVR Workshop*, pages 313–321. OCG-Schriftenreihe, Österr. Arbeitsgemeinschaft für Mustererkennung, R. Oldenburg, 1995. Band 81.

[18] David Marr. Visual information processing: The structure and creation of visual representations. *Philosophical Transactions of the Royal Society London*, Ser. B(290):pp.199–218, 1980.

[19] J. Matas and Josef Kittler. Spatial and feature space clustering: Applications in image analysis. In Václav Hlaváč and Radim Šára, editors, *Proceedings CAIP'95, Computer Analysis of Images and Patterns*, volume Vol. 970 of *Lecture Notes in Computer Science*, pages 162–173, Prague, Czech Republic, 1995. Springer, Berlin Heidelberg, New York.

[20] Christophe Mathieu, Isabelle E. Magnin, and C. Baldy-Porcher. Optimal stochastic pyramid: segmentation of MRI data. *Proc. Med. Imaging VI: Image Processing*, SPIE Vol.1652:pp.14–22, Feb. 1992.

[21] Peter Meer. Stochastic image pyramids. *Computer Vision, Graphics, and Image Processing*, Vol. 45(No. 3):pp.269–294, March 1989.

[22] Peter Meer, Song-Nian Jiang, Ernest S. Baugher, and Azriel Rosenfeld. Robustness of image pyramids under structural perturbations. *Computer Vision, Graphics, and Image Processing*, Vol. 44(No. 3):pp.307–331, December 1988.

[23] Annick Montanvert and Pascal Bertolino. Irregular pyramids for parallel image segmentation. In Walter Kropatsch and Horst Bischof, editors, *Pattern Recognition 1992*, OCG-Schriftenreihe, Österr. Arbeitsgemeinschaft für Mustererkennung. R. Oldenburg, 1992. Band 62.

[24] Peter F.M. Nacken. Image segmentation by connectivity preserving relinking in hierarchical graph structures. *Pattern Recognition*, 28(6):907–920, June 1995.

[25] Theo Pavlidis. *Structural Pattern Recognition*. Springer, New York, 1977.

[26] Markus Stricker and Aleš Leonardis. ExSel++, a General Framework to Extract Parametric Models. In *Proceedings CAIP'95, Computer Analysis of Images and Patterns*, volume Vol. 970 of *Lecture Notes in Computer Science*, pages 90–97, Prag, Czech Republic, 1995. Springer.

[27] Eiichi Tanaka. A Metric between Unrooted and Unordered Trees and its Bottom-Up Computing Method. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(12):1233–1238, December 1994.

[28] Dieter Willersinn. Parallel Graph Contraction for Dual Irregular Pyramids. Technical Report PRIP-TR-28, Institute f. Automation 183/2, Dept. for Pattern Recognition and Image Processing, TU Wien, Austria, 1994.