Building Irregular Pyramids by Dual Graph Contraction*

Walter G. Kropatsch

Abstract

Many image analysis tasks lead to or make use of graph structures that are related through the analysis process with the planar layout of a digital image. This paper presents a theory that allows to build different types of hierarchies on top of such image graphs. The theory is based on the properties of a pair of dual image graphs that the reduction process should preserve, e.g. the structure of a particular input graph. The reduction process is controlled by decimation parameters, i.e. a selected subset of vertices, called survivors, and a selected subset of the graph's edges, the parent-child connections. It is formally shown that two phases of contractions transform a dual image graph to a dual image graph built by the surviving vertices. Phase one operates on the original (neighborhood) graph and eliminates all non-surviving vertices. Phase two operates on the dual (face) graph and eliminates all degenerated faces that have been created in phase one. The resulting graph preserves the structure of the survivors, it is minimal and unique with respect to the selected decimation parameters. The result is compared with two modified specifications, the one already in use for building stochastic and adaptive irregular pyramids.

*This work was supported by the Austrian Science Foundation under grant S 7002-MAT

1 Introduction

The need for hierarchies in image analysis has been expressed by many scientists, e.g. recently by Nagy [14]. Multiresolution pyramids are already widely used in image analysis [15, 4, 19]. Hierarchies are motivated both by biological plausibility [18] and by computational efficiency [7].

Adjacency plays an important role in image analysis, too. Starting with the definition of neighboring pixels in low level processes up to adjacencies defined between regions resulting from segmentation processes, graphs can be used to represent these adjacency concepts. Although regular neighborhood structures dominate the lower levels of image processing and other data structures like arrays may be more efficient, at later processing stages regularity cannot be imposed.

Irregular pyramids combine graph structures with hierarchies. Similar to regular pyramids, we distinguish ordered levels of decreasing sizes in an irregular pyramid. Each level is a graph describing the image. Adjacent levels in decimation pyramids are related by the fact that the vertex set of the reduced level is a subset of the vertices in the level below. The methods for building irregular pyramids differ in several aspects:

1. in the way they select the survivors;

2. in the way they derive the neighborhood relations of the reduced level.

The first aspect may heavily depend on the kind of application. A typical application is in the field of image segmentation [13], for an overview over different graph theoretical approaches to clustering and segmentation see [20]. Also regular pyramids fit into this general framework: Their survivors are predetermined and form a regular pattern. Regular pyramids suffer from the rigidity of their structure that causes sensitivity to pixel shifts and artefacts when used for segmentation [3] or for the analysis of line drawings [8]. The abandonned regularity constraints in irregular pyramids allow random selections as used in stochastic pyramids [12], but also very sophisticated methods that adapt the new structure to the data such as adaptive pyramids [6]. But one could also imagine selection criteria that are influenced by a certain processing goal. Our approach decouples selection and contraction by clearly specifying the decimation parameters that control the reduction and by requiring a few constraints that these parameters should satisfy (see Section 3.1).

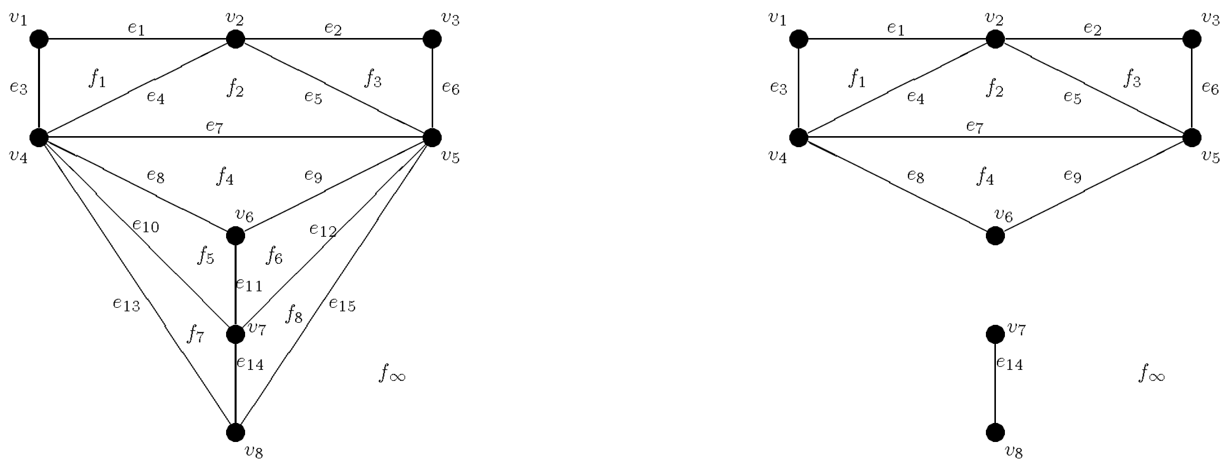
The second aspect allows several variations. Rosenfeld [16] has related parallel, degree-preserving graph contraction to multiresolution techniques. The framework he presents for parallel contraction operations depends on algebraic properties of regular graphs like trees, hypercubes, arrays, etc. Our theory extends the scope of parallel, degree-preserving graph contraction to irregular topologies. We define "connecting paths" that relate the edges of the reduced graph with paths between surviving vertices in the level below. The basic operation that contracts the graphs either step-by-step or in a few parallel steps is dual contraction. It contracts one edge and its two endpoints into one single vertex and removes the corresponding dual edge. The contraction of a graph reduces the number of vertices while maintaining the connections to other vertices. As a consequence self-loops and double edges may occure. The elimination of such non-simple connections may lead to configurations that corrupt the connectivity structure given in the input graph. We shall overcome these problems by considering the dual graph.

The remainder of this paper is organized as follows. Section 2 recapitulates the basic notions from graph theory and introduces the concept of dual image graphs. Considering crossing of paths and interior vertices we define the structure of a graph. Based on this framework, we define what we mean by a structure preserving contraction (Section 3). Dual graph contraction proceeds in two phases, dual edge contraction and dual face contraction. Both of these two operations are defined and their respective properties discussed in subsections 3.2 and 3.3 respectively. The introduced concepts are illustrated by means of simple examples. Section 4 compares the structural properties of three related ways to build irregular pyramids. The conclusion summarizes the results, offers several possibilities for selecting the decimation parameters and for reducing the information stored in the cells of the pyramid.

2 Dual image graphs and their structure

This section assembles the terminology from graph theory that is needed to define the type of graphs and the notations that describe a structure in a digital image.

We use *graphs* $G(V, E)$ consisting of *vertices* $v \in V$ and (non-directed) *edges* $e \in E$. An edge e connects two vertices v, w, $e_i = (v, w)$, an edge with $v = w$ is called a *self-loop*: $e_i(v, v)$. A graph may contain more than one edge between the same end vertices (i.e. $e_3(v_1, v_8) \neq e_5(v_1, v_8)$ in Fig. 8a), they are called *double edges*[1]. Edges are uniquely identified by indices. The *degree* of a vertex v, $\deg(v)$, is the number of edges *incident* on it. A vertex $v \in V$ is *isolated* if it has degree 0, i.e. $\deg(v) = 0$. Formal definitions of standard notions are taken from [17, 5], here, a simple example explains the basic terms.



(a) Graph $G_1(V_1, E_1)$ before and (b) after removal of cutset.

Figure 1: Graph $G_1(V_1, E_1)$ is disconnected by cutset $\{e_{10}, e_{11}, e_{12}, e_{13}, e_{15}\}$.

Figure 1 shows a graph $G_1(V_1, E_1)$, with vertices $V_1 = \{v_1, \ldots, v_8\}$ and edges $E_1 = \{e_1, \ldots, e_{15}\}$. Edge $e_1(v_1, v_2)$ connects vertices v_1 and v_2. The degree of vertex v_5 is six, e.g. $\deg(v_5) = 6$, since the six edges $e_5, e_6, e_7, e_9, e_{12}, e_{15}$ are incident to v_5. *Path*

[1]Another name is *parallel edge*.

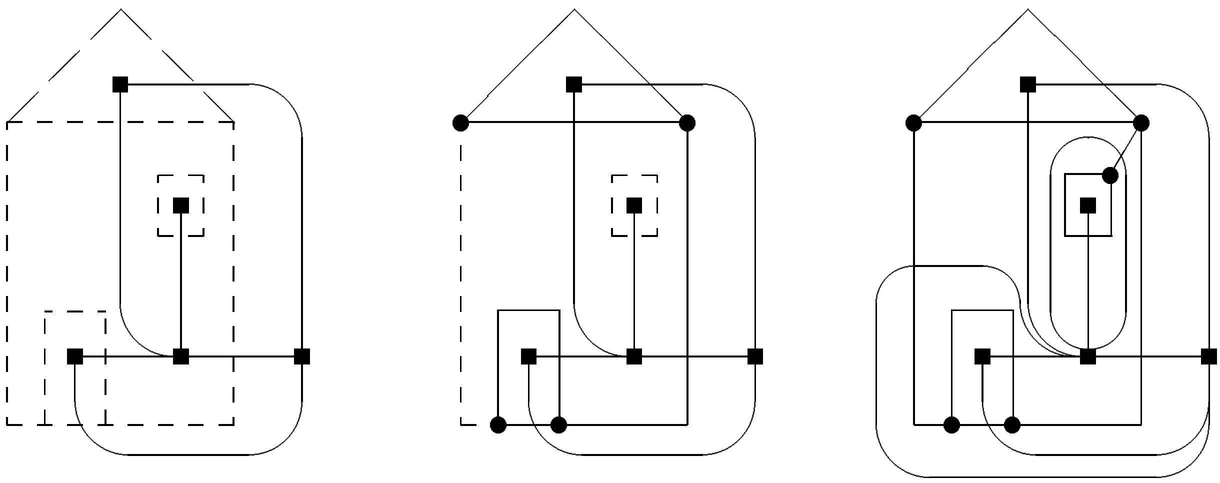
$P_{63}(v_6, v_3) = (v_6, e_{11}, v_7, e_{12}, v_5, e_6, v_3)$ connects v_6 with v_3 traversing three edges. It has *length* three, $\|P_{63}\| = 3$, the same length as path $P_{18}(v_1, v_8) = (v_1, e_1, v_2, e_5, v_5, e_{15}, v_8)$. The *circuit* $C_1 = (v_8, e_{13}, v_4, e_7, v_5, e_{15}, v_8)$ in Fig. 1(a) is a closed path in G_1. Since any pair of vertices of G_1 can be connected by a path in G_1, graph G_1 is *connected.* If edges are removed from E_1, the graph may become disconnected. After removal of $E_c = \{e_{10}, e_{11}, e_{12}, e_{13}, e_{15}\}$, graph $G_1'(V_1, E_1 \setminus E_c)$ is disconnected (Fig. 1(b)) and consists of two *connected components* $\{v_1, v_2, v_3, v_4, v_5, v_6\}$ and $\{v_7, v_8\}$. The subset of edges $E_c \subset E_1$ is called a *cutset.*

Graph $G_1(V_1, E_1)$ is *planar* since it is drawn in the plane without any edge crossing another edge. A graph can be embedded in the plane in many ways. A graph already embedded in the plane is called a *plane graph.* The planar embedding of G_1 in Fig. 1(a) divides the plane into 8 (finite) regions which are called *faces*, $f_1, \ldots, f_8$, and one infinite region, the *background face* f_∞. A *cycle* $C(f)$ delimits exactly one face f, e.g. $C(f_3) = (v_2, e_5, v_5, e_6, v_3, e_2, v_2)$. The *boundary* of a (finite) graph is the cycle delimiting the background face, $C_\infty := C(f_\infty) = (v_1, e_1, v_2, e_2, v_3, e_6, v_5, e_{15}, v_8, e_{13}, v_4, e_3, v_1)$. The adjacency of the faces in G_1 is expressed by the dual graph, $\overline{G_1}(\overline{V_1}, \overline{E_1})$, Fig. 7(b). There exists a one-to-one correspondence between the edges $\overline{e_i}$ of $\overline{G_1}$ and the edges e_i of G_1. Furthermore, any set of edges is a circuit in $\overline{G_1}$ if and only if the corresponding set of edges is a cutset in G_1. E.g. the edges corresponding to cutset $E_c \subset E_1$ form a circuit $(\overline{v_7}, \overline{e_{10}}, \overline{v_5}, \overline{e_{11}}, \overline{v_6}, \overline{e_{12}}, \overline{v_8}, \overline{e_{15}}, \overline{v_\infty}, \overline{e_{13}}, \overline{v_7})$ in $\overline{G_1}$.

2.1 Graphs of images

Our graphs describe the neighborhood relations in a digital image. At low level processing, a pixel of the sensor array is associated with a vertex and pixels adjacent either in a row or in a column are joined by an edge (note that we use 4-connectivity). The gray value or any more complex description is considered as an attribute of a vertex but is not directly used in the algorithms of this paper. The resulting graphs have several properties, they are finite, connected, and plane. We consider both the *neighborhood graph* $G(V, E)$ and its dual graph $\overline{G}(\overline{V}, \overline{E})$ in parallel. Since the vertices of $\overline{G}$ are the faces of G we refer to $\overline{G}$ as the *face graph.* This pair of related graphs is the basis of all further considerations.

The same graph formalism as for the pixel array can be used also at intermediate levels of image analysis: Region adjacency graphs (RAGs) are the result of segmentation processes. Regions are connected sets of pixels, two regions are separated by the common boundaries. Although RAGs are connected since the regions partition the image plane, their geometric duals may cause problems. Consider the RAG, $\overline{G_2}(\{\blacksquare\}, \{\blacksquare - -\blacksquare\})$, of the



(a) RAG $\overline{G_2}$ (b) $G_2 = dual(\overline{G_2})$ (c) DIG $(G_2', \overline{G_2'})$

Figure 2: A house: (a) RAG $\overline{G_2}$; (b) reconstructed G_2; (c) corrected DIG $(G_2', \overline{G_2'})$.

house example in Fig. 2(a). The five regions of the house, e.g. roof, window, door, front side, and background, are indicated by dashed lines. To reconstruct the boundary graph G_2, i.e. the dual of $\overline{G_2}$, we insert a vertex ($\bullet$) in each region of $\overline{G_2}$ and place them on the dashed boundary, preferably at boundary intersections. Then we draw the edges of E_2 by following the dashed boundary lines until crossing an edge of $\overline{E_2}$ (similar to [5][p.113]). Two problems arise in this case:

1. The window is completely surrounded by the region of the front side. Hence its boundary is not connected with the boundary of the front side. Where to place the vertex of V_2? If placed as shown in Fig. 2(b) the above algorithm terminates but

does not find any edge crossing the window boundary. In the other placement the algorithm does not find any correct solution.

2. The left hand boundary of the front side is not crossed by any edge of $\overline{E_2}$.

The problems are caused by the fact that the front side's boundary consists of two non-connected pieces: the inner piece common with the window, and the outer piece being further split into four segments: one segment separates it from the roof, another from the door, and two distinct segments separate it from the background. In fact graph $\overline{G_2}$ does not express that the window is completely within the front side and that the door creates the two distinct boundary segments separating it from the background. A solution is shown in Fig. 2(c): a self-loop around the window is added in $\overline{G'_2}$, front side and background are connected by a double edge in $\overline{E'_2}$, and a 'bridge' edge in G'_2 connects the boundary of the window with the boundary of the front side. The resulting pair of graphs are connected and plane, but, unfortunately, in general not simple. E.g. they may contain self-loops and double edges. However not all possible self-loops and double edges are necessary. The necessary cases can be limited to those where the self-loop or the double edges enclose non-neglectable details like the window or the door in the above example. Redundant configurations will be characterized by degenerated vertices in the dual graph (section 3.3). The following definition summarizes the properties of dual image graphs.

Definition 1 (Dual Image Graphs) *The graphs* $(G(V,E), \overline{G}(\overline{V},\overline{E}))$ *are called* ***dual image graphs*** *(DIGs) if they have the following properties:*

- *both* G *and* $\overline{G}$ *are finite;*
- *both* G *and* $\overline{G}$ *are connected;*
- *both* G *and* $\overline{G}$ *are plane;*
- $\overline{G}$ *is the dual of* G*;*
- *both* G *and* $\overline{G}$ *need not be simple in general.*

2.2 The structure of plane graphs

The structure of an image plays a fundamental role in image analysis because it is invariant to any 2D image transformation and because it allows to identify objects in images by their topological structure. But what do we mean by structure precisely? We have encountered already several properties that characterize a structure and that allow to disambiguate different structures.

The two paths P_{63} and P_{18} in graph G_1 of Fig. 1(a) *intersect* at vertex v_5. More formally we define whether two paths cross each other in a given graph.

Definition 2 (Crossing Paths) *Let P_1 and P_2 be two paths in a plane graph $G(V,E)$ with a common path $P_0 \subset P_1 \cap P_2$, such that $P_1 = (P_{1a}, P_0, P_{1b})$ and $P_2 = (P_{2a}, P_0, P_{2b})$. P_0 can be as short as only one single vertex. Path P_1 crosses path P_2 if the four path tails alternate in a clockwise enumeration around P_0, e.g. $(P_{1a}, P_{2a}, P_{1b}, P_{2b})$ (Fig. 3).*

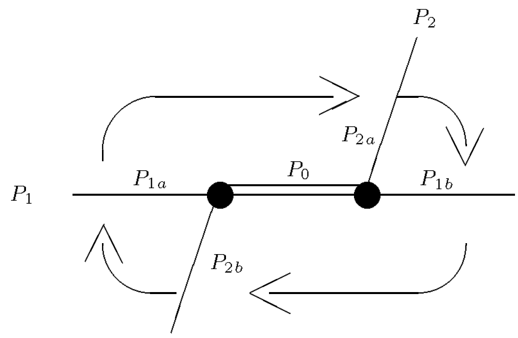


Figure 3: Crossing of two paths P_1 and P_2.

A substructure like a single vertex, a single face, a subgraph, ... that is completely surrounded by a circuit contributes also to the structure. Remember the window in the house example. In Fig. 1(a) circuit $C_1 = (v_8, e_{13}, v_4, e_7, v_5, e_{15}, v_8)$ completely surrounds vertex v_6. We call v_6 *interior vertex* of C_1 and define this relation between a single vertex and a circuit in a plane graph as follows:

Definition 3 (Interior Vertex) *Let $C \neq C_\infty$ be a circuit in a finite, connected, plane graph $G(V, E)$. Furthermore, let C_∞ denote the cycle delimiting the background of G. A vertex $v \in V$ is called an* ***interior vertex of*** *C if there is no path $P(v, v_\infty)$ connecting v to any vertex $v_\infty \in C_\infty \setminus C$ without crossing C, e.g. $P(v, v_\infty) \cap C \neq \emptyset$ (Fig. 4). Circuit C is said to 'surround' vertex v.*

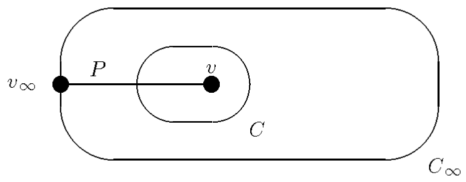


Figure 4: Vertex v is interior of circuit C.

We describe an image's adjacency relations by a pair of plane graphs. The formal definition of the structure of a plane graph collects all the above determining factors.

Definition 4 (Structure of a Plane Graph) *Let $G(V, E)$ be a finite, connected, plane graph. Furthermore, let $S_G(v)$ denote the family of all circuits surrounding vertex $v \in V$ in graph G. Then we define as the structure of G the following set:*

$$Struct(G) := \{(v, S_G(v)) | v \in V\}$$

This definition captures the topological properties present in a DIG. As an example recall Fig. 2c). The boundary between the wall and the background consists of two distinct parts separated by the door. This fact is expressed in a pixel representation by two disjoint sequences of edges and in the corrected DIG by a double edge between the corresponding vertices. The preservation of an image's structure facilitates the recognition of objects by their structure in a very condensed description.

3 Dual Graph Contraction

In this section we present the algorithm that simplifies the structure of a pair of dual image graphs. The contraction process is controlled by decimation parameters. Selected subsets of vertices and of edges of the original neighborhood graph define the relation between the contracted and the original graphs. Subsection 3.1 specifies the required properties of the contracted graphs. The structure modification consists of two elementary operations described in subsections 3.2 and 3.3 that are combined in the algorithm in subsection 3.4.

3.1 Structure preserving contraction

Stochastic decimation as proposed by Meer [12] is controlled by selecting surviving and non-surviving vertices, and by defining receptive fields that completely cover the input data. Jolion and Montanvert [6] showed how this selection must be modified such that decimation is controlled by the image data in order to achieve an adaptive behavior of the process.

Definition 5 (Decimation Parameters) *Consider a graph $G(V, E)$. A decimation of graph G is specified by a selection of* ***surviving*** *vertices $V_s \subset V$ and a selection of a subset E_{sn} of edges E. The sets (V_s, E_{sn}) are called* ***decimation parameters****. We call $V_n := V \setminus V_s$* ***non-surviving vertices****. E_{sn} must be a subset of $(V_s \times V_n) \cap E$ and it connects all non-surviving vertices to exactly one surviving vertex in a unique way:*

$$\forall v_n \in V_n \qquad \exists! v_s \in V_s \qquad \exists! e \in E_{sn} \qquad e = (v_s, v_n) \tag{1}$$

Subgraph (V, E_{sn}) partitions G into the same number of connected components as there are surviving vertices in V_s. Each component forms a tree structure connecting the surviving vertex, the **parent** ($\bullet$), to the non-surviving vertices, the **children** ($\circ$), by edges of E_{sn} ($\bullet \rightarrow \circ$, see example in Fig. 5).

Note that our definition does not constrain the selection of surviving vertices, as does the requirement of a maximum independent set (MIS) in stochastic pyramids [12]. Adaptive

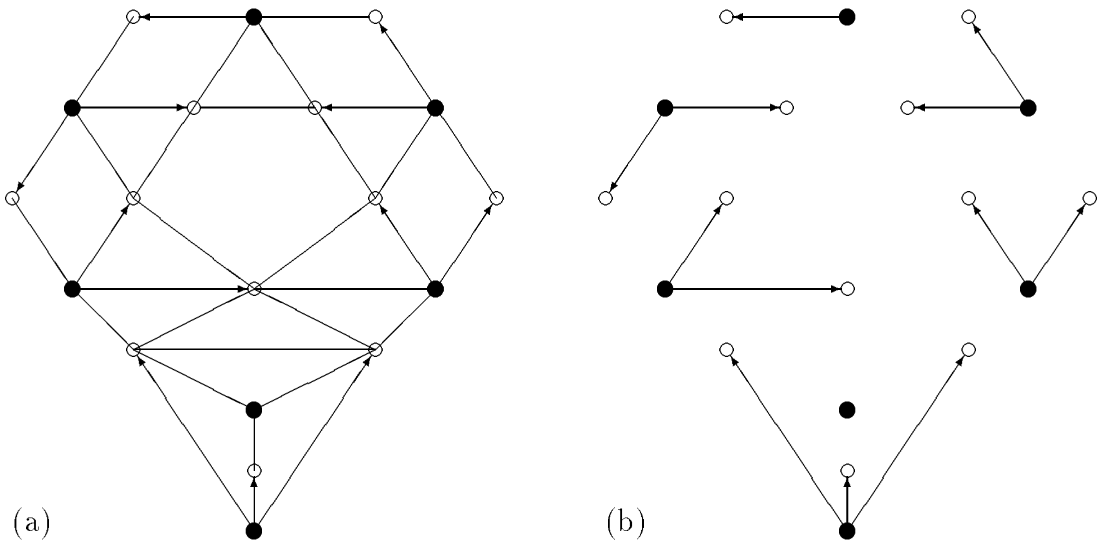


Figure 5: Decimation of $G_3(V_3, E_3)$ creates trees in graph (V_3, E_{sn}).

decimations as in [6] can be uniformly treated with the above definition. Only non-surviving vertices must have a surviving neighbor.

Before defining the properties that characterize a contracted graph $G'(V', E')$ we introduce *connecting paths* in $G(V, E)$ that relate edges of E' with paths in G.

Definition 6 (Connecting Path) *Let G(V,E) be a graph with decimation parameters (V_s, E_{sn}). A path in $G(V, E)$ is called a **connecting path** of two surviving vertices $v_b, v_e \in V_s$, denoted $CP(v_b, v_e)$, if one of the following conditions is satisfied:*

1. *v_b and v_e are connected by an edge e_{be} in G: $CP(v_b, v_e) = (v_b, e_{be}, v_e); e_{be} \in E$.*
2. *The path contains two edges, $CP(v_b, v_e) = (v_b, e_{bi}, v_i, e_{ie}, v_e)$ with $v_i \in V_n$ and one of the two edges is in E_{sn}.*
3. *The path contains three edges of E, $CP(v_b, v_e) = (v_b, e_{bi}, v_i, e_{ij}, v_j, e_{je}, v_e)$ with both $v_i, v_j \in V_n$ and both edges $e_{bi}, e_{je} \in E_{sn}$.*

Connecting paths have lengths 1, 2, or 3. The end points of connecting paths are surviving vertices. Every connecting path contains exactly one edge that is not in E_{sn}. Connecting paths are the basis to define neighbors in the contracted graph.

Definition 7 (Structure Preserving Contraction) *Graph $G'(V', E')$ is a **structure preserving contraction** of a connected, plane graph $G(V, E)$ controlled by decimation parameters (V_s, E_{sn}) if following conditions are satisfied:*

1. *$V' = V_s$.*
2. *For all edges $e' = (v_b, v_e) \in E'$ there exists a connecting path $CP(v_b, v_e)$ in G.*
3. *If $CP(v_b, v_e)$ is a connecting path in G then $v_b = v_e$ or $(v_b, v_e) \in E'$.*
4. *Let C be any sequence of connecting paths $CP(v_0, v_1), CP(v_1, v_2), \ldots, CP(v_n, v_0)$ in G forming a circuit. If there exist surviving vertices interior of C they must also be interior of the circuit $C' = (v_0, (v_0, v_1), v_1, \ldots (v_n, v_0), v_0)$ in G'.*

The first three conditions establish the correspondence between graph $G(V, E)$ and the contracted graph $G'(V', E')$. The selected survivors V_s are the vertices of the contracted graph V'. Edges in E' correspond to connecting paths in G and vice versa, and, consequently, circuits in G' have corresponding circuits in G. Circuit C in the fourth condition characterizes all circuits in G that have a corresponding circuit in G'. Let $v_s \in V_s$ be surrounded by C, then $C \in S_G(v_s)$ (cf. Def.4). Condition 4 requires that any 'surviving' part (v_s, C) of the structure of G is preserved in the structure of G', e.g. $C' \in S_{G'}(v_s)$. Since this must be true for all circuits C' surrounding v_s in G', $(v_s, S_{G'}(v_s)) \in Struct(G')$.

3.2 Dual contraction of non-surviving vertices

Two vertices v_i and v_j in a graph $G(V, E)$ are **identified** by replacing both vertices by a new vertex which is connected to all vertices that were incident on v_i and v_j before identification. **Contraction** of an edge $e \in E$ in a graph $G(V, E)$ is the operation of removing e from E and identifying its end vertices [17].

Definition 8 (Dual Edge Contraction) *Let $G(V,E)$ and $\overline{G}(\overline{V},\overline{E})$ be dual image graphs.* ***Dual contraction*** *contracts an edge $e \in E$ and removes its corresponding edge $\overline{e} \in \overline{E}$ from $\overline{G}$ at the same time.*

Theorem 1 *Let $G(V,E)$ and $\overline{G}(\overline{V},\overline{E})$ denote dual image graphs and (V_s, E_{sn}) the decimation parameters. Dually contracting all edges of E_{sn} collapses all non-surviving vertices into their surviving parents and creates a contracted graph $G'(V',E')$ that preserves the structure of $G(V,E)$ (according to Def. 7). All connecting paths become edges of the contracted graph $G'(V',E')$ connecting the surviving endpoints.*

The proof of this theorem can be found in [10].

The above process can be implemented in parallel for two reasons: (1) because the removal of edges E_{sn} from E and $\overline{E_{sn}}$ from $\overline{E}$ is independent of each other and (2) because identification simply renames all children to their parents' name in the remaining sets E and $\overline{E}$.

3.3 Dual contraction of redundant faces

Dual edge contraction of graph $G(V,E)$ decreases the number of edges in $\overline{E}$ and, hence, also the degrees of the vertices in $\overline{G}$. Faces with degree one and two may result. They correspond to self-loops and double edges in the neighborhood graph, they do not surround any surviving vertex and, hence, they do not contribute to the structure of the graph. A second (dual) contraction process 'cleans' the dual graph from such **degenerated** faces.

Definition 9 (Dual Face Contraction) *Consider a pair of dual image graphs $G(V,E)$ and $\overline{G}(\overline{V},\overline{E})$. Let $\overline{v_i} \in \overline{V} \setminus \{\overline{v_\infty}\}$ be a degenerated face not being the background face, $\deg(\overline{v_i}) < 3$, and let $\overline{e_i}(\overline{v_i}, \overline{v_j})$ be an incident edge in $\overline{E}$. Then $\overline{e_i}$ is dually contracted, identifying $\overline{v_i}$ with $\overline{v_j}$, and eliminating edge $e_i \in E$ corresponding to $\overline{e_i}$. Since vertices of $\overline{G}$ correspond to faces of G, we refer to this process as* ***dual face contraction.***

Theorem 2 *Let $G(V,E)$ and $\overline{G}(\overline{V},\overline{E})$ be a pair of dual image graphs. Dual face contraction preserves the structure of graph $G(V,E)$.*

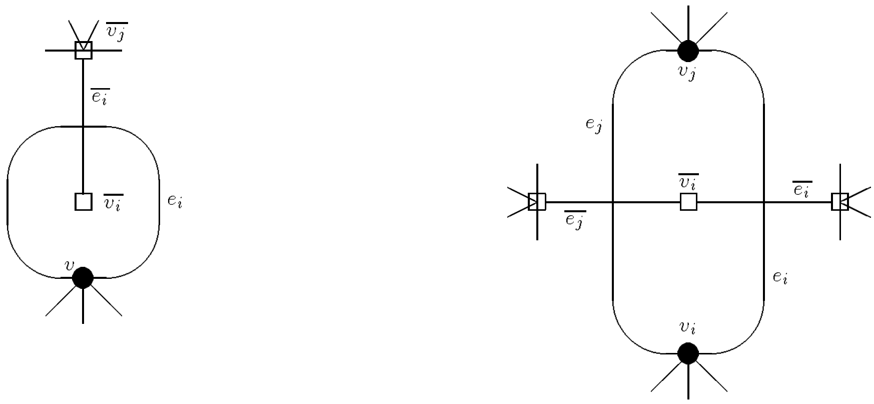


Figure 6: Face contraction of degenerated faces.

Proof : The two cases $\deg(\overline{v_i}) = 1$ and $\deg(\overline{v_i}) = 2$ (see Fig. 6) are discussed separately:

1. If $\deg(\overline{v_i}) = 1$ then the edge $e_i \in E$ corresponding to $\overline{e_i} = (\overline{v_i}, \overline{v_j})$ is the only edge in the circuit surrounding face f_i, i.e. $e_i = (v, v)$ is a self-loop in G. Clearly the removal of a self-loop does not disconnect G. Self-loops in G that contain interior vertices are not removed because any interior non-isolated vertex would increase $\deg(\overline{v_i}) > 1$.

2. Let $\deg(\overline{v_i}) = 2$, $\overline{e_i}, \overline{e_j} \in \overline{E}$ being the two edges incident to $\overline{v_i}$. If $\overline{e_i} = \overline{e_j}$, $\overline{e_i} = (\overline{v_i}, \overline{v_i})$ is a self-loop in $\overline{G}$. Since $\overline{G}$ is connected, $\overline{E} = \{\overline{e_i}\}$ and $\overline{V} = \{\overline{v_i}\}$, $\overline{v_i}$ being the only face. Hence $\overline{v_i} = \overline{v_\infty}$ is the background face which is excluded from face contraction. Therefore $\overline{e_i} \neq \overline{e_j}$ are different edges in $\overline{G}$ and, by duality, $e_i \neq e_j$ are different edges in G. Face $\overline{v_i}$ is surrounded by a circuit C with two edges in G: $C = (v_i, e_i, v_j, e_j, v_i)$. Obviously both e_i and e_j connect the same vertices v_i and v_j in G. The removal of one of such double edges preserves the connectivity of G. C does not contain any interior vertex and any other circuit surrounds the same vertices before and after contracting $\overline{v_i}$. ■

Note that the contraction of a face may lead to another degenerated face. Furthermore, not all degenerated faces can be contracted in parallel. However a process similar to stochastic decimation can determine an independent set of degenerated faces which could be contracted in parallel. For the remaining degenerated faces the process is repeated until no further degenerated face exists in $\overline{G}$.

3.4 Combining the elementary processes

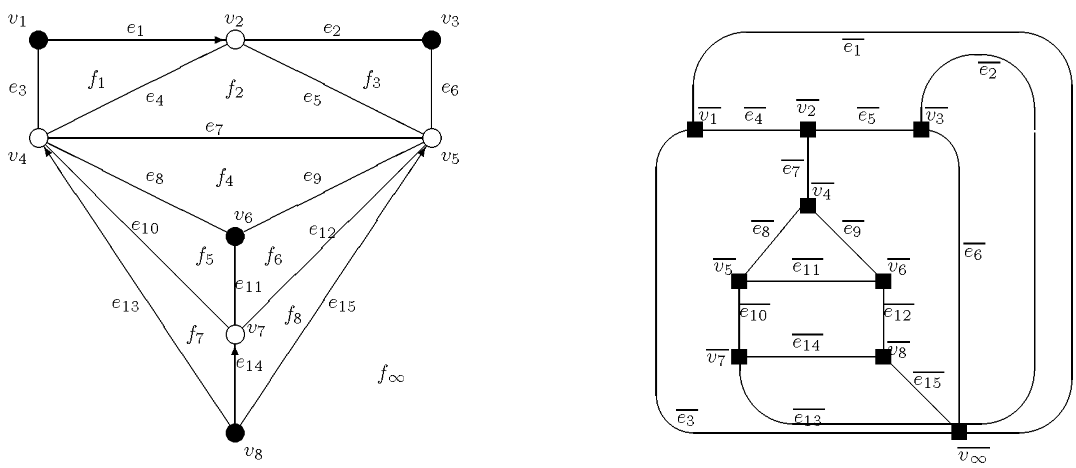
In the previous subsections, we have gathered all subprocesses we need to define the process of dual graph contraction.

Definition 10 (Dual Graph Contraction) *Let $G(V, E)$ and $\overline{G}(\overline{V}, \overline{E})$ be a pair of dual image graphs. Given the decimation parameters (V_s, E_{sn})* ***dual graph contraction*** *consists of the following sequence of processes applied to this pair of graphs:*

1. *Dually contract all edges $e \in E_{sn}$ collapsing all non-surviving vertices into their surviving parent vertex;*

2. *dually (face) contract all degenerated faces;*

3. *repeat step 2 until all degenerated faces have been eliminated.*

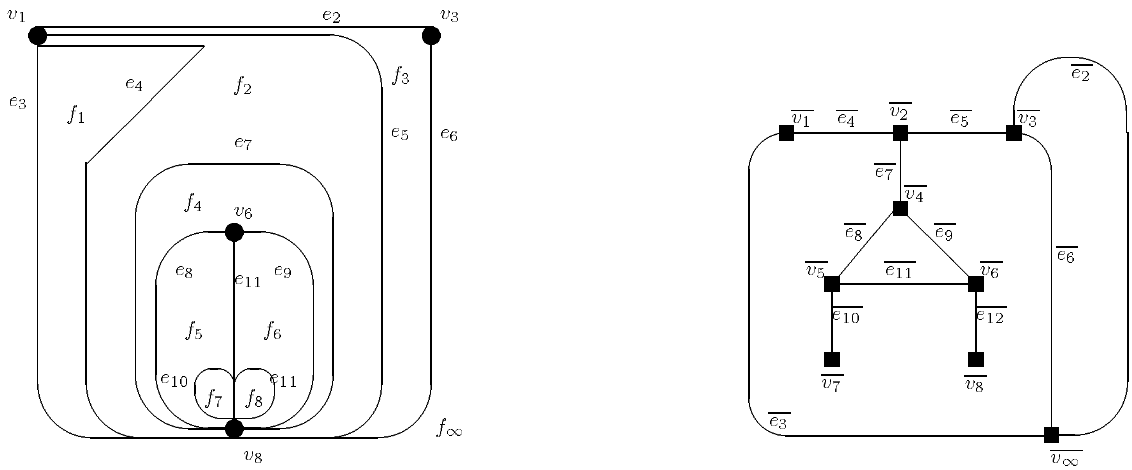


(a) G_1 with decimation parameters (b) dual graph $\overline{G_1}$

Figure 7: G_1 and $\overline{G_1}$ before dual graph contraction.

Figures 7, 8 and 9 illustrate dual graph contraction. Figure 7 shows a planar embedding of graph $G_1(V_1, E_1)$ consisting of 8 vertices and 15 edges. The plane is divided into 9 faces, with face f_∞ being the background face. Figure 7b) shows the dual of G_1, $\overline{G_1}(\overline{V_1}, \overline{E_1})$, with one vertex representing every face of G_1. Note that all finite faces form triangles, or equivalently, $\deg(\overline{v_i}) = 3, i = 1, \ldots, 8$. Figure 7a) illustrates the decimation parameters graphically: survivors $V_s = \{\bullet\}$, non-survivors $V_n = \{\circ\}$, and $E_{sn} = \{\bullet \rightarrow \circ\} = \{e_1, e_{13}, e_{14}, e_{15}\}$.

The result of dually contracting all edges E_{sn} is depicted in Figure 8: all parent-child

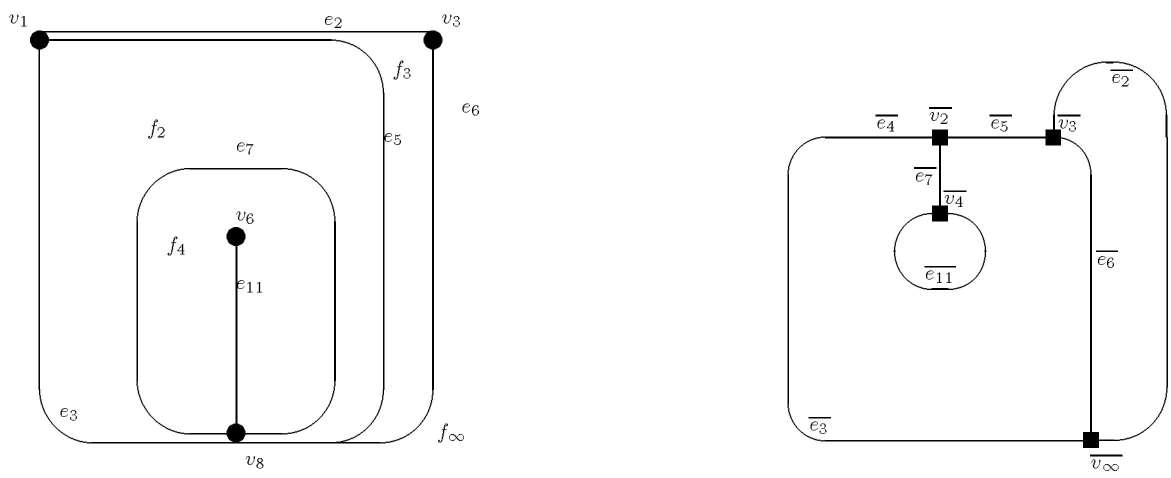


(a) G_1^* (b) $\overline{G_1^*}$.

Figure 8: Result of dual edge contraction: G_1^* and $\overline{G_1^*}$.

connections, i.e. all edges that are drawn as arrows in Figure 7a), have been dually contracted. Graph [2] G_1^* contains three self-loops: e_7, e_{10}, e_{11}. There are three edges connecting the same two end vertices v_1 and v_8: e_3, e_4, e_5; and also three edges connecting v_6 and v_8: e_8, e_9, e_{11}. Fig. 9 results from dually contracting all degenerated faces in $\overline{G_1^*}$ successively: f_7, f_8, f_1, f_5, f_6. Note that v_6 is interior both to circuit $(v_1, e_1, v_2, e_5, v_5, e_{15}, v_8, e_{13}, v_4, e_3, v_1)$ in G which becomes $(v_1, e_5, v_8, e_3, v_1)$ in G', and to $(v_8, e_{13}, v_4, e_7, v_5, e_{15}, v_8)$ in G becoming a self-loop (v_8, e_7, v_8) in G'. Hence self-loop e_7 as well as double edge e_3, e_5 must survive

[2] G^* identifies the result of dual edge contraction, G' the final result.



(a) G'_1 (b) $\overline{G'_1}$

Figure 9: Result of dual graph contraction: G'_1 and $\overline{G'_1}$.

to satisfy condition (4) of Def. 7.

Theorem 3 *Let $(G(V,E), \overline{G}(\overline{V}, \overline{E}))$ be a pair of dual image graphs and $(G'(V',E'), \overline{G'}(\overline{V'}, \overline{E'}))$ be the result of dual graph contraction with decimation parameters (V_s, E_{sn}). Then*

1. *$G'(V',E')$ is a structure preserving contraction of $G(V,E)$.*

2. *$(G'(V',E'), \overline{G'}(\overline{V'}, \overline{E'}))$ is minimal, i.e. no further contraction is possible.*

3. *$(G'(V',E'), \overline{G'}(\overline{V'}, \overline{E'}))$ is unique.*

Proof :

1. $(G'(V',E'), \overline{G'}(\overline{V'}, \overline{E'}))$ is a structure preserving contraction of $G(V,E)$ since all the involved operations, e.g. dual edge contraction and dual face contraction, preserve the structure given in $G(V,E)$, as proved in Theorems 1 and 2.

   However, connectivity of $\overline{G'}$ has not been shown yet. Connectivity of graph $\overline{G}$ is preserved when degenerated faces are contracted. Therefore, $\overline{G}$ could be disconnected only by dual edge contraction. Let $e_i \in E_{sn}$ be the last edge the dual contraction of which would split $\overline{G}$ into two components. Hence $\overline{e_i}$ is the only connection between the two parts before splitting and, as a consequence, e_i must be a self-loop in G. This contradicts the assumption that $e_i \in E_{sn}$ connects a surviving with a non-surviving vertex.

2. $(G'(V', E'), \overline{G'}(\overline{V'}, \overline{E'}))$ is minimal if any further contraction would destroy the desired properties of G'. Since $V' = V_s$, no further vertex can be removed by dual edge contraction. After step (3) of dual graph contraction there are no degenerated faces other than the background face in $\overline{G'}$, e.g. $\deg(\overline{v_i}) > 2$ for all $\overline{v_i} \in \overline{V'} \setminus \{\overline{v_\infty}\}$. Let us consider the consequences of dually contracting a face with $\deg(\overline{v_i}) > 2$ by dually contracting an incident edge $\overline{e_{ij}} = (\overline{v_i}, \overline{v_j})$. The removal of edge e_{ij} from G' would either disconnect two vertices that were connected before (and, hence, were also connected in G by a connecting path) or it would open a circuit build by a double edge that surrounds a surviving subgraph. This substructure exists because otherwise the double edge would include a degenerated face with only two sides.

3. $G'(V', E')$ is unique. It is clear that the result after step (1) is unique since it removes all non-surviving vertices and since the individual operations are independent.

   Now let us assume we have derived two different results $(G'_1(V'_1, E'_1), \overline{G'_1}(\overline{V'_1}, \overline{E'_1}))$ and $(G'_2(V'_2, E'_2), \overline{G'_2}(\overline{V'_2}, \overline{E'_2}))$ by dually contracting the faces in a different order. $V'_1 = V'_2$ because face contraction does not change vertices. The connectivity in G' is determined by the connecting paths and not by the order of face contractions. Hence also $E'_1 = E'_2$ and $G'_1 = G'_2$. The dual graphs may differ only by a different planar embedding. But this is determined by the structure of the graphs before dual contraction and preserved by dual face contraction. ∎

4 Three different ways to build irregular pyramids

Def. 7 specified four properties that relate the original graph G and its contraction G'. It was argued that these conditions should preserve certain structural properties of graph G.

With some slight modifications of the requirements other results can be achieved. This section compares the introduced version with two modifications. In the examples we shall use graph G_3 from Fig. 5(a) as our original graph.

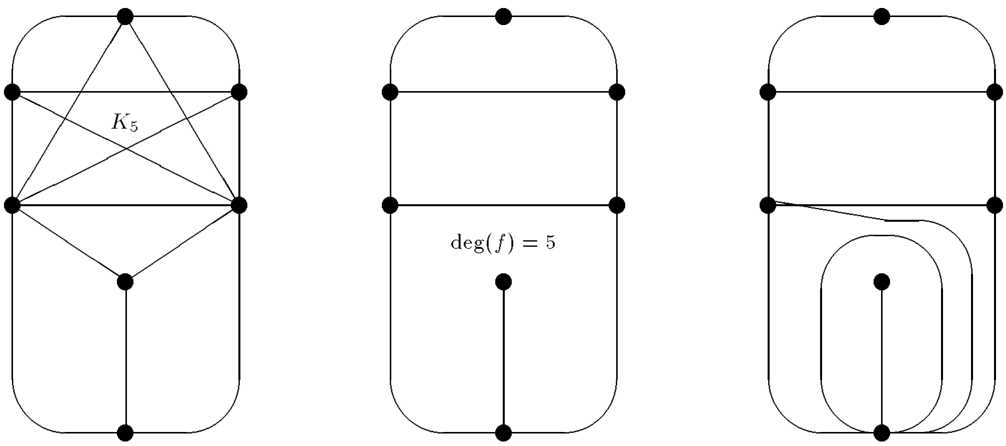
The following property has been observed first in [11] but the present formulation allows a much clearer proof.

Theorem 4 *Let* $(G(V,E),\overline{G}(\overline{V},\overline{E}))$ *be a pair of dual image graphs and* $(G'(V',E'),\overline{G'}(\overline{V'},\overline{E'}))$ *the result of dual graph contraction. Then the degree of vertices of* $\overline{G'}$ *is less or equal to the degree of vertices of* $\overline{G}$*.*

Proof : Dual edge contraction removes dual edges in $\overline{G}$, but the number of faces remains the same as in $\overline{V}$. However the degrees of the two adjacent faces decrease by one when a dual edge is removed. Dual face contraction eliminates degenerated faces. Contraction of a face of degree one reduces the degree of the other adjacent face by one. Contraction of a face with degree two leaves the degrees of the two adjacent faces the same. Hence all faces of $\overline{G'}$ can find a face in $\overline{G}$ with at least the same degree. ∎

If we relaxe the requirement to preserve structure (fourth condition in Def. 7), the resulting graphs need no self-loops nor any double edges. The minimal graph satisfying conditions (1), (2), and (3) is *simple*, it has been used in the previous works of Meer [12], Montanvert [13] and Jolion [6]. The such defined simple graph is a subgraph of a structure preserving contraction. Since the structure preserving contraction preserves planarity this is also the case for the simple graph. Let us refer to this type of contraction as *simple contraction*. The only drawback of the simple contraction is that the degrees of faces cannot be garanteed to shrink in certain cases, e.g. when there exists a vertex with degree one (see Fig. 10b).

The second modification further relaxes the definition of connecting paths. In simple contraction connecting paths are not longer than 3, but not all paths of lengths less than four are connecting paths. This last extension allows all such paths to create an edge in the reduced graph G': $E' := \{(u,v) \in V_s \times V_s | \exists P(u,v) \in G \text{ such that } \|P(u,v)\| < 4\}$. Let us refer to this graph reduction as *path length contraction* although it does not necessarily involve a contraction operation. With this simplification, the selection of E_{sn} is no more necessary. In addition, planarity cannot be preserved. This is illustrated in Figure 10 which shows the result of the three different contractions of the same graph G_3 shown in Fig. 5. Although the original graph is planar the graph in Fig. 10(a) contains the complete graph K_5 as subgraph. Table 1 compares properties of the three contractions.



(a) path length (b) simple (c) structure preserving

Figure 10: Three different contractions of G_3.

5 Conclusion

Dual graph contraction transforms a pair of dual image graphs into a pair of smaller dual image graphs. The contraction is controlled by decimation parameters. Surviving vertices can be chosen as an arbitrary subset of vertices, only non-surviving vertices must satisfy a minor constraint. It is shown that the result preserves the structure given before contraction. Furthermore it fulfills all requirements for dual image graphs to be contracted again. Applied recursively, the algorithm builds an irregular pyramid.

Table 1: Comparison of three contractions $G(V,E) \rightarrow G'(V',E')$.

path length	simple	structure preserving
$V' := V_s$	$V' := V_s$	$V' := V_s$
$\|CP(u,v)\| < 4$	$CP(u,v)$ (Def. 6)	$CP(u,v)$ (Def. 6)
$(u,v) \in E' \Leftrightarrow \exists \|P(u,v)\| < 4 \in G$	$(u,v) \in E' \Leftrightarrow CP(u,v) \in G$	(1)–(4) of Def. 7
no double edge	no double edge	some double edges
no self-loop	no self-loop	some self-loops
planar $\rightarrow$ non-planar	planar $\rightarrow$ planar	planar $\rightarrow$ planar
connected $\rightarrow$ connected	connected $\rightarrow$ connected	connected $\rightarrow$ connected
preserving lengths of cycles(?)	not preserving face degrees	preserving face degrees

Our experience with the different approaches for reducing graph structures and the new approach presented in this paper extends the scope of the presented theory to three and higher dimensions. We observed that contraction led to degenerations both in the original and in the dual graph (self-loops, double edges). When removed in the original graph the structure could not be preserved. However the removal of degenerations in the dual graph nicely removed all degenerations that did not destroy the structure of the graph. In 3D space duality can be introduced between points and volumes, and between lines and faces. A similar dual contraction scheme could be applied to build 3D irregular pyramids.

How to select the decimation parameters has not been discussed in this paper. There are several possibilities to determine these parameters, each criterium following a different objective:

- Random selection as in Meer's stochastic pyramids [12];
- MIS determination by a Hopfield neural network [2, 1];
- Adapting the pyramid structure by data dependent local voting like in [6];

- Enforcing certain model-guided subgraph structures that could be predetermined by the vocabulary of interpretation.

Besides their structural information the vertices and edges of dual image graphs carry additional information as do the pixels of a picture array. Semantic information can be added to the graphs by attributes or labels. During contraction these attributes must be calculated also for the reduced graph. In analogy to regular pyramids reduction functions [9] serve this purpose. They take as input the attributes of all children to compute the parent's attribute. Subsampling or averaging would be simple examples. The real potential of irregular pyramids lies probably in the efficient combination of reducing information and adaptively contracting the structure.

Acknowledgments

The author would like to thank the reviewers and following colleagues and students who contributed to the ideas presented in this paper at various stages of development: Dieter Willersinn, Herwig Macho, Etienne Bertin, Peter Nacken, and Horst Bischof. The final revision of the paper was done during the author's sabbatical at LIP-ENS Lyon[3] the support of which is greatfully acknowledged.

References

[1] Etienne Bertin and Horst Bischof. Voronoi Pyramids controlled by Hopfield networks. Technical Report PRIP-TR-24, Technical University of Wien, Dept. Pattern Recognition and Image Processing 183/2, TU Wien, Austria, 1993.

[2] Horst Bischof and Walter G. Kropatsch. Hopfield networks for irregular decimation. In Wolfgang Pölzleitner and Emanuel Wenger, editors, *Image Analysis and Synthe-*

[3] Ecole Normale Superieure de Lyon, France

sis, pages 317–327. OCG-Schriftenreihe, Österr. Arbeitsgemeinschaft für Mustererkennung, R. Oldenburg, 1993. Band 68.

[3] 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.

[4] Virginio Cantoni and Stefano Levialdi, editors. *Pyramidal Systems for Image Processing and Computer Vision*, volume F25 of *NATO ASI Series*. Springer-Verlag Berlin, Heidelberg, 1986.

[5] F. Harary. *Graph Theory*. Addison-Wesley, Reading, Mass., 1972.

[6] 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.

[7] Jean-Michel Jolion and Azriel Rosenfeld. *A Pyramid Framework for Early Vision*. Kluwer Academic Publishers, 1994.

[8] Walter G. Kropatsch. Curve Representations in Multiple Resolutions. *Pattern Recognition Letters*, Vol. 6(No. 3):pp.179–184, August 1987.

[9] Walter G. Kropatsch. Irregular pyramids. In Peter Mandl, editor, *Modelling and New Methods in Image Processing and in Geographical Information Systems*, OCG-Schriftenreihe, Österr. Arbeitsgemeinschaft für Mustererkennung, pages 39–50. Oldenbourg, 1991. Band 61.

[10] 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.

[11] Walter G. Kropatsch and Annick Montanvert. Irregular versus regular pyramid structures. In U. Eckhardt, A. Hübler, W. Nagel, and G. Werner, editors, *Geometri-*

cal Problems of Image Processing, pages 11–22, Georgenthal, Germany, March 1991. Akademie Verlag, Berlin.

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

[13] Annick Montanvert, Peter Meer, and Azriel Rosenfeld. Hierarchical image analysis using irregular tesselations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-13(No.4):pp.307–316, April 1991.

[14] George Nagy. An Induced Taxonomy of Visual Form. In Carlo Arcelli, Luigi P. Cordella, and Gabriella Sanniti di Baja, editors, *2nd Intl. Workshop on Visual Form*, Capri, Italy, June 1994.

[15] Azriel Rosenfeld, editor. *Multiresolution Image Processing and Analysis*. Springer, Berlin, 1984.

[16] Azriel Rosenfeld. Arc colorings, partial path groups, and parallel graph contractions. Technical Report TR-1524, University of Maryland, Computer Science Center, July 1985.

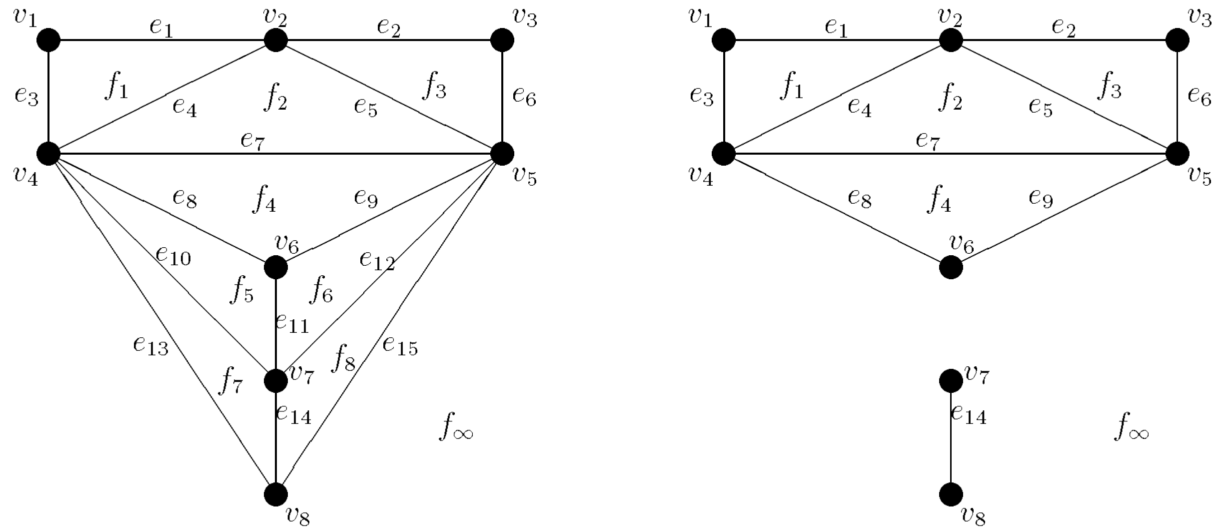
[17] K. Thulasiraman and M.N.S. Swamy. *Graphs: Theory and Algorithms*. Wiley-Interscience, 1992.

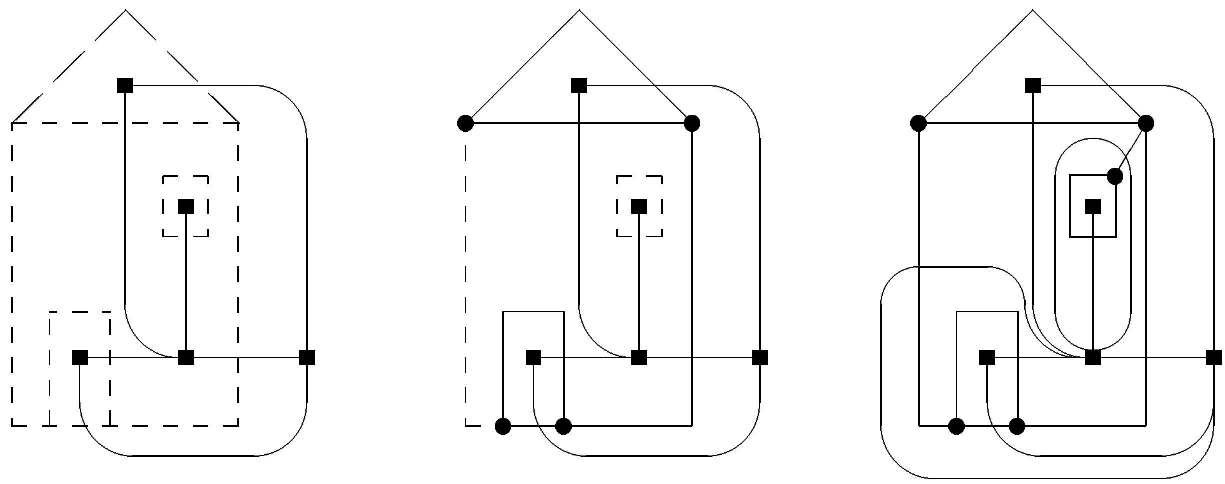
[18] John K. Tsotsos. Analyzing vision at the complexity level. *Behavioral and Brain Sciences*, 13(3):423–469, 1990.

[19] Leonard Uhr, editor. *Parallel Computer Vision*. Academic Press Inc., New York, 1987.

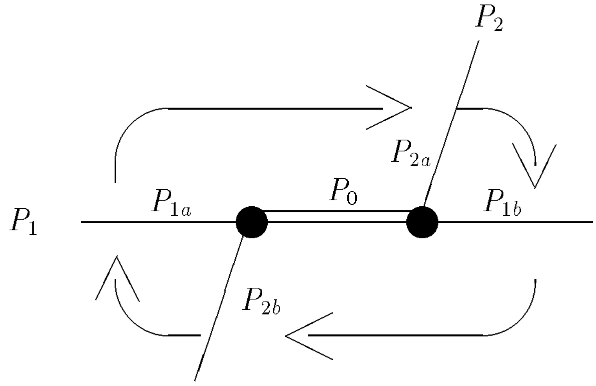
[20] Zhenyu Wu and Richard Leahy. An Optimal Graph Theoretic Approach to Data Clustering: Theory and Its Application to Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.15(11):pp. 1101–1113, November 1993.

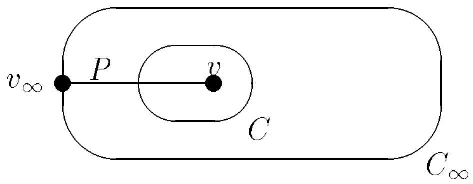
List of Figures

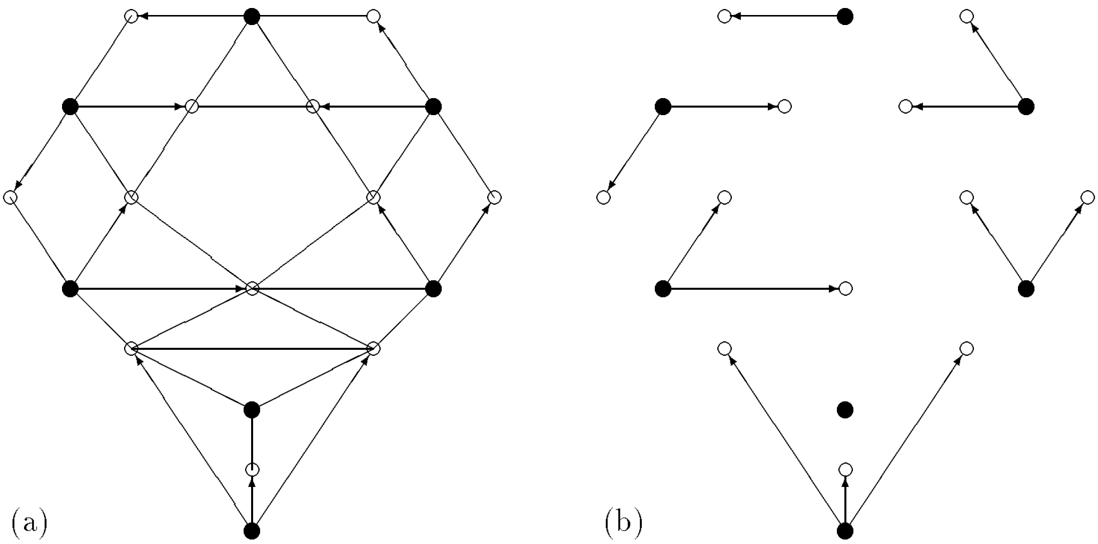


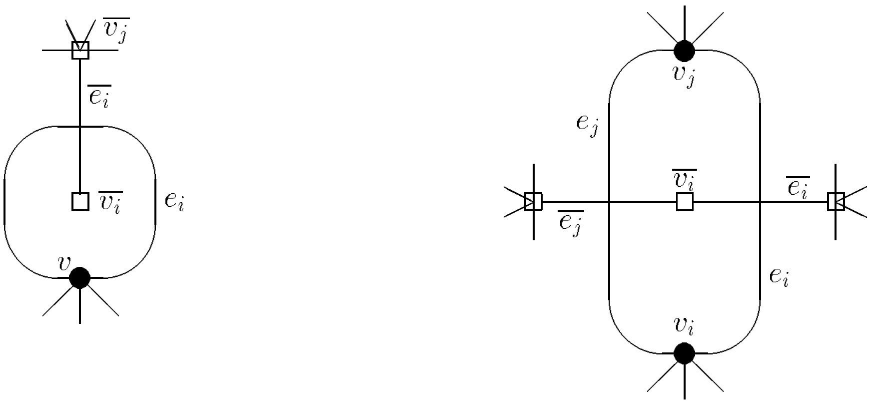


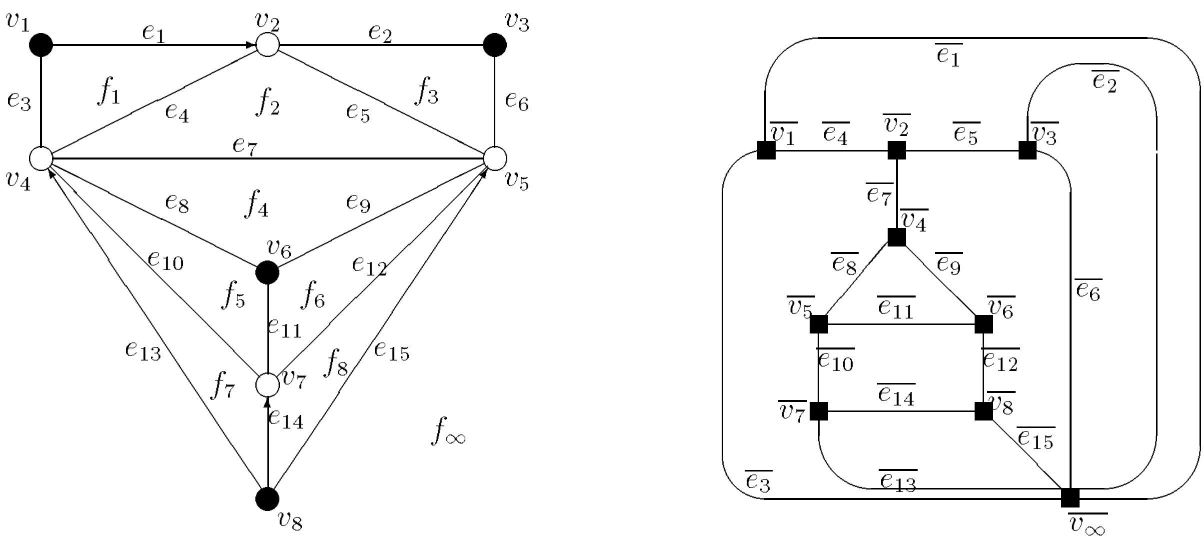
(a) RAG $\overline{G_2}$ (b) $G_2 = dual(\overline{G_2})$ (c) DIG $(G_2', \overline{G_2'})$

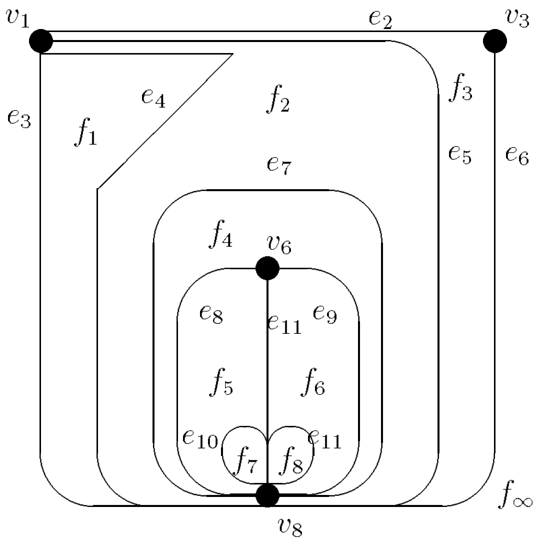

P_2
P_{2a}
P_1
P_{1a}
P_0
P_{1b}
P_{2b}

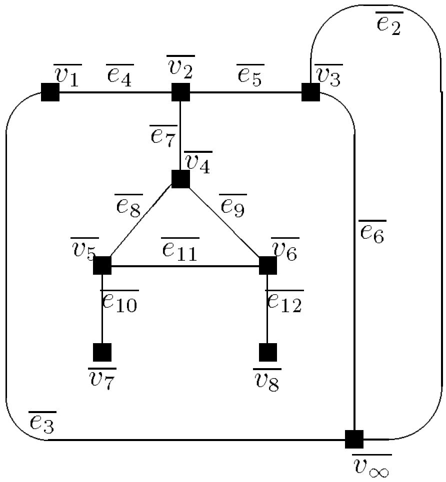

v_∞
P
v
C
C_∞

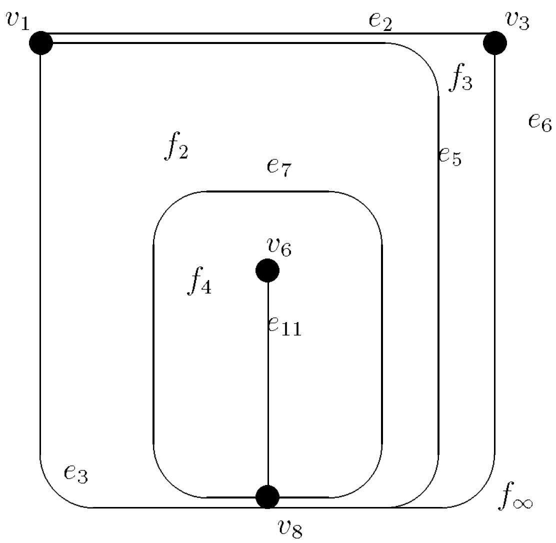

(a)
(b)

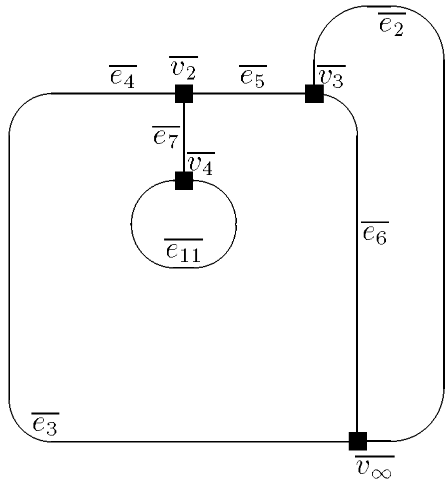

$\overline{v_j}$
$\overline{e_i}$
$\overline{v_i}$
e_i
v
v_j
e_j
$\overline{v_i}$
$\overline{e_i}$
$\overline{e_j}$
e_i
v_i

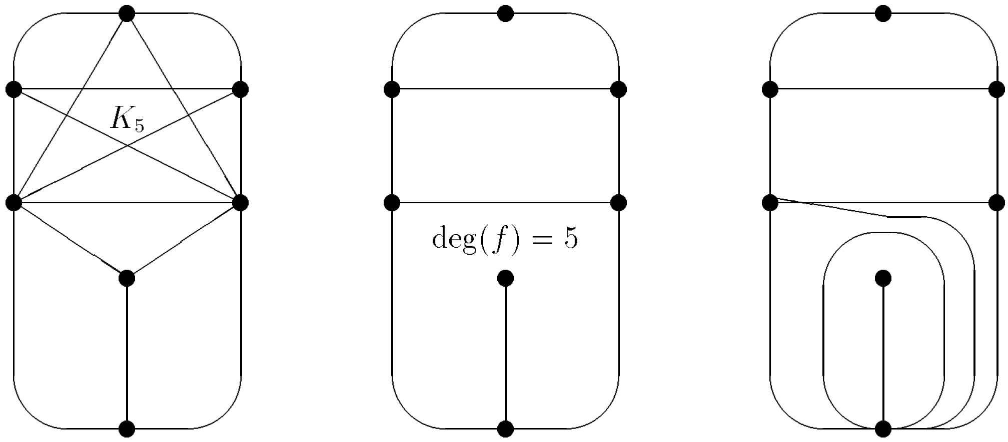

v_1 e_1 v_2 e_2 v_3
e_3 f_1 e_4 f_2 e_5 f_3 e_6
e_7
v_4 v_5
e_8 f_4 e_9
v_6
e_{10} e_{12}
f_5 f_6
e_{11}
e_{13} f_8 e_{15}
f_7 v_7
e_{14}
f_∞
v_8
$\overline{e_1}$
$\overline{e_2}$
$\overline{v_1}$ $\overline{e_4}$ $\overline{v_2}$ $\overline{e_5}$ $\overline{v_3}$
$\overline{e_7}$
$\overline{v_4}$
$\overline{e_8}$ $\overline{e_9}$
$\overline{e_6}$
$\overline{v_5}$ $\overline{e_{11}}$ $\overline{v_6}$
$\overline{e_{10}}$ $\overline{e_{12}}$
$\overline{v_7}$ $\overline{e_{14}}$ $\overline{v_8}$
$\overline{e_{15}}$
$\overline{e_3}$ $\overline{e_{13}}$
$\overline{v_\infty}$


v_1
e_2
v_3
e_3
f_1
e_4
f_2
f_3
e_5
e_6
e_7
f_4
v_6
e_8
e_{11}
e_9
f_5
f_6
e_{10}
e_{11}
f_7
f_8
f_∞
v_8


$\overline{e_2}$
$\overline{v_1}$
$\overline{e_4}$
$\overline{v_2}$
$\overline{e_5}$
$\overline{v_3}$
$\overline{e_7}$
$\overline{v_4}$
$\overline{e_8}$
$\overline{e_9}$
$\overline{e_6}$
$\overline{v_5}$
$\overline{e_{11}}$
$\overline{v_6}$
$\overline{e_{10}}$
$\overline{e_{12}}$
$\overline{v_7}$
$\overline{v_8}$
$\overline{e_3}$
$\overline{v_\infty}$


v_1
e_2
v_3
f_3
e_6
f_2
e_5
e_7
v_6
f_4
e_{11}
e_3
f_∞
v_8


$\overline{e_2}$
$\overline{e_4}$
$\overline{v_2}$
$\overline{e_5}$
$\overline{v_3}$
$\overline{e_7}$
$\overline{v_4}$
$\overline{e_{11}}$
$\overline{e_6}$
$\overline{e_3}$
$\overline{v_\infty}$



(a) path length (b) simple (c) structure preserving