Hierarchical Image Analysis Using Irregular Tessellations

Annick Montanvert, *Member, IEEE,* Peter Meer, *Member, IEEE,* and Azriel Rosenfeld, *Fellow, IEEE*

***Abstract*—We present a novel multiresolution image analysis technique based on hierarchies of irregular tessellations generated in parallel by independent stochastic processes. Like the "traditional" image pyramids these hierarchies are constructed in on the order of log(image_size) steps. However, the structure of a hierarchy is adapted to the image content and artifacts of rigid resolution reduction are avoided. We give two applications of our technique: connected component analysis of labeled images, and segmentation of gray level images. In labeled images, every connected component is reduced to a separate root, with the adjacency relations among the components also extracted. In gray level images the output is a segmentation of the image into a small number of classes as well as the adjacency graph of the classes.**

***Index Terms*—Connected components, image pyramids, image segmentation, irregular tessellations, multiresolution techniques.**

I. Introduction

A CELLULAR pyramid is an exponentially tapering stack of arrays of processors ("cells"). Communication between cells on successive levels of the stack allows global analysis of data input to the base of the stack in log(base_size) steps. Cellular pyramids support fast parallel algorithms for multiresolution image analysis. See the books edited by Rosenfeld [31], Cantoni and Levialdi [8], and Uhr [37]; for solving computational geometry problems [25], with the image input to level 0, the base of the pyramid.

Usually the cells on each level are connected to form a square lattice but triangular or hexagonal grids have also been used [2], [6], [14]. A cell on level $l + 1$ (the *parent*) is connected to a $K \times K$ neighborhood of cells on level l (its *children*). Neighborhoods associated with adjacent parents overlap by K–2 cells along both directions, yielding a fourfold reduction in number of processors (twofold along each side of the square); however, twofold reductions can also be achieved using modified architectures [11], [18].

We restrict ourselves here to pyramids defined on a square grid with fourfold reduction between successive levels. If an image is input to the base of the pyramid, we can generate reduced-resolution versions of the image at higher levels. Usually the value of the parent is a weighted average of the values of its children and the same set of weights is employed at every level. Burt [7] defined rules for which the set of weights converges to sampled Gaussians with increasing standard deviations. Optimal weights have also been proposed [24].

Let the base of the pyramid be of size $N^2 = 2^n \times 2^n$. Then the lth level has size $2^{n-l} \times 2^{n-l}$, so that the total number of cells is less than $\frac{4}{3}N^2$. The height of the pyramid, i.e., the number of levels, is $n = \log N$. Many image analysis tasks which require $O(N^2)$ operations on a single processor can be accomplished in $O(\log N)$ on a cellular pyramid.

When a pyramid is used to reduce the resolution of an image, features of the input image become smaller and move closer together as one proceeds from the bottom level of the pyramid to its apex. Thus at the appropriate level, local operations are sufficient to detect and analyze global features (see [31] for numerous examples). Reduced resolution representations are also useful in image compression applications (e.g., [1]).

The case of $K = 2$, i.e., nonoverlapping 2×2 neighborhoods, is related to the quadtree description of an image [34]. The reduced resolution representations can be severely distorted when the input is shifted [36]. This problem is known in the quadtree literature as the *shift-dependence* of the description [20]. In the worst case a one pixel shift of the input image can lead to a significantly modified quadtree structure [35].

The dependence of the low resolution representations on the position of the sampling grid and the input image is also important in image pyramid applications. The shift-dependence phenomenon is not restricted to the case of nonoverlapping neighborhoods. Bister [5] shows many examples of such artifacts.

The rigidity of the pyramid structure may give rise to artifacts when pyramids are used for tasks such as analysis of line-drawings [19], object-background discrimination [10], or compact object extraction [13], [16]. To compensate for these artifacts, in many of these algorithms the parent–child links (or link weights) are iteratively changed after the initial resolution reduction stage. Recently Baronti *et al.* [4] proposed a modification of this concept by increasing the size of the neighborhoods associated with parents once an initial segmentation of the image is obtained.

Another approach to compensating for the artifacts of pyramid structure is to adapt this structure to the content of the input image. In custom-made pyramids [28] weights are defined based on a local "busyness" measure during the construction of the reduced resolution representations. Rom and Peleg [29] and Chassery and Montanvert [9] employed the Voronoi tessellation defined by a set of randomly chosen lattice points to build the coarsest representation of the image, which was then adaptively refined. Note that the method computes the representations top-to-bottom.

In this paper we also use irregular tessellations to generate an adaptive multiresolution representation of the input image. In our approach, however, the hierarchy of representations is built bottom-up and is adapted to the content of the input image; thus most of the properties of "classical" image pyramids are

Manuscript received October 1, 1989; revised June 14, 1990. Recommended for acceptance by C. Brown. This work was performed while A. Montanvert was visiting the Center for Automation Research. A. Montanvert was supported by a NATO Fellowship. P. Meer and A. Rosenfeld were supported by the National Science Foundation under Grant DCR-86-03723 and by the U.S. Air Force Office of Scientific Research under Grant AFOSR-86-0092.

A. Montanvert is with Equipe RFMQ-TIM3, Universite Joseph Fourier Grenoble II, P. O. Box 53x, 38041 Grenoble Cedex, France.

P. Meer is with the Department of Electrical and Computer Engineering, Rutgers University, New Brunswick, NJ 08903.

A. Rosenfeld is with the Center for Automation Research, University of Maryland, College Park, MD 20742.

IEEE Log Number 9041538.

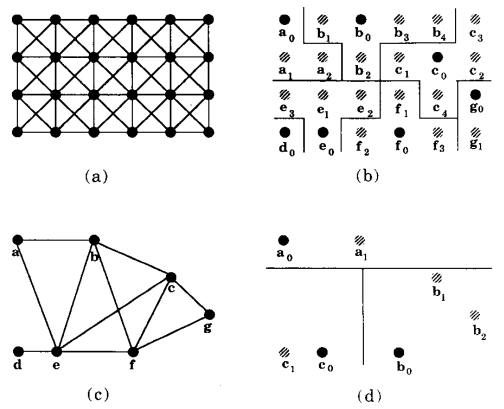

(a)
(b)
(c)
(d)
a b c d e f g
a0 b1 b0 b3 b4 c3 a1 a2 b2 c1 c0 c2 e3 e1 e2 f1 c4 g0 d0 e0 f2 f0 f3 g1
a0 a1 b1 b2 c1 c0 b0

Fig. 1. Example of an irregular tessellation hierarchy which does not satisfy our constraints. See text for discussion. (a) The eight-connected square lattice as a graph. (b) An arbitrary partition of the graph. Filled circles: survivors retained for the next level. Hashed circles: nonsurvivors. (c) The graph of the next level. (d) An arbitrary partition of (c).

preserved. We employ a local stochastic process to build the lower resolution representations.

In Section II we introduce the graph formulation of irregular tessellations and the concept of a stochastic image pyramid. In Sections III and IV we give two applications of stochastic pyramids: connected component analysis of labeled images and segmentation of gray-scale images. Further issues are discussed in Section V.

II. Irregular Sampling and Stochastic Pyramids

In image pyramids based on regular sampling, e.g., at points on a square grid, artifacts caused by the rigidity of the sampling structure are always present. On the other hand, an image pyramid defined by an irregular sampling hierarchy can be molded to the structure of the input image. Note, however, that in such a pyramid the metrical relations among cells are no longer carried implicitly by the sampling structure. A cell at level $l + 1$ cannot know *a priori* where its neighbors on level $l + 1$ or its children on level l are located relative to the original sampling grid. To describe the structure of such an image pyramid it is more appropriate to use the formalism of graphs.

A. Graph Representation of Irregular Sampling Hierarchies

The cells on level l of the pyramid are taken as the vertices of an undirected graph $G[l]$. The edges of the graph describe the adjacency relations between cells at level l. Thus $G[l] = (V[l], E[l])$ where $V[l]$ is the set of vertices and $E[l]$ is the set of edges. The graph $G[0]$ defined by the 8-connected square sampling grid on level 0 is shown in Fig. 1(a). An example of a graph $G[1]$ that might represent level 1 is shown in Fig. 1(c).

We construct the pyramid by a sampling or decimation process. Each level is constructed from the level below it by selecting a subset of the vertices. Thus a vertex on any level can be regarded as a vertex of $G[0]$, the sampling grid of the original image. In addition, when we decimate level l to construct level $l + 1$, we associate each nonsurviving vertex with one of the surviving vertices. Thus each vertex on level $l + 1$ is associated with a set of vertices on level l (itself and the nonsurviving vertices associated with it). Each of these vertices is in turn associated with a set of vertices on level $l - 1$, and so on; thus a vertex on any level is associated with a set of vertices, called its "region," in the original image. These regions define a tessellation of the image.

If the pyramid is to be build recursively bottom-up we must define a procedure for deriving $G[l + 1]$ from $G[l]$. Since the number of vertices in $G[l + 1]$ must be less than in $G[l]$ we are dealing with a *graph contraction* problem. We must design rules for:

- choosing from the set of vertices $V[l]$ the new set $V[l+1] \subset V[l]$, i.e., the survivors of the decimation process;
- allocating each nonsurviving vertex of level l to a survivor, i.e., generating the parent–children links;
- creating the edges $E[l + 1]$, i.e., defining the adjacency relations among the surviving vertices of level l.

In order to have any vertex (i.e., cell) in the hierarchy correspond to a connected region of the image, the cell $c[l + 1] \in V[l + 1]$ must represent a connected subset of cells $\{c_0[l], c_1[l], \cdots, c_u[l]\} \subset V[l]$. We shall use the convention $c_0[l] \equiv c[l + 1]$, i.e., the surviving vertex of the subset is first on the list. In pyramid terminology, $\{c_0[l], c_1[l], \cdots, c_u[l]\}$ are the children of $c[l + 1]$. Note that the location of the parent on the sampling grid of the original image always coincides with the location of one of its children.

In pyramid construction based on Voronoi tessellations [9], [29] the parents are initially chosen by a random process. The edges are given by the Delaunay diagram of the tessellation and the children are grid sites inside the tiles associated with the parents. The process can then be repeated for individual tiles (by randomly choosing grid sites inside each tile) to obtain a finer description. Note that such a pyramid is built top-down and the definitions of parents and parent–children links are based on nonlocal processes.

When we use graph contraction to construct a pyramid, two constraints must be satisfied if we want to employ only parallel local processes:

$$\forall\, i = 1, 2, \cdots, u \qquad (c_i[l], c_0[l]) \in E[l] \tag{1}$$
$$\forall\, c_0[l] \in V[l],\; d_0[l] \in V[l] \qquad (c_0[l], d_0[l]) \notin E[l] \tag{2}$$

where c_0, d_0 are survivors on level l. Constraint (1) assures that any nonsurvivor on cell at level l has at least one survivor in its neighborhood and thus can be allocated to a parent by a local decision. In the example shown in Fig. 1(b) this constraint is satisfied. Constraint (2) assures that two adjacent cells on level l cannot both survive and thus the number of vertices must decrease rapidly from level to level. In Fig. 1(b) this constraint is not satisfied since the survivors d_0, e_0 and c_0, g_0 are adjacent. The construction of $G[l + 1]$ can also be regarded as finding a maximal collection of vertices of $G[l]$ no two of which are adjacent. This is the maximal independent set problem for graphs (e.g., [21]); we will return to it in Section V.

A possible alternative method of constructing $G[l + 1]$ from $G[l]$ is to partition $G[l]$ into connected subgraphs and then select one cell in each subgraph as a survivor. However, if we do so, the first constraint no longer assures locality of the processing. In Fig. 1(b) cell b_4 has survivor c_0 adjacent to it, but must be allocated to survivor b_0 two sites away. Choosing the survivor independently for each region may also violate the second constraint since two adjacent regions can both have their survivors at the border [Fig. 1(b)]. Thus the set of children should be defined in $G[l]$ only after the vertices of $G[l + 1]$ (their parents) have been chosen.

The last step in constructing $G[l+1]$ is to define the edges $E[l+1]$. Let the connected subsets $\{c_0[l], c_1[l], \cdots, c_u[l]\} \subset V[l]$ and $\{d_0[l], d_1[l], \cdots, d_v[l]\} \subset V[l]$ be the children of two different parents. Our condition for an edge between vertices $c[l+1] \equiv c_0[l]$ and $d[l+1] \equiv d_0[l]$ in $G[l+1]$ is

$$\exists\, 0 \le i \le u,\ 0 \le j \le v \qquad (c_i[l], d_j[l]) \in E[l]. \tag{3}$$

In other words, two vertices are joined by an edge in $G[l+1]$ if there exists a path between them in $G[l]$ of length at most three edges. (Note that by (2), the path cannot be of length 1.) $G[l+1]$ is now completely defined. Fig. 1(c) shows, the graph corresponding to the partition in Fig. 1(b).

The irregular sampling hierarchy is thus built recursively from $G[0]$ (the original sampling grid). The apex of the hierarchy $G[m]$ has only one vertex. Constraint (2) assures that the apex is always reached.

In the next section we describe a probabilistic parallel algorithm that constructs graph contractions satisfying (1) and (2). The algorithm is analyzed in more detail in [22], [23].

B. Stochastic, Data-Dependent Decimation

We have seen that the derivation of $G[l+1]$ from $G[l]$ must start by defining the vertices of the new graph. Since $V[l+1] \subset V[l]$ we are dealing with a *decimation* process, i.e., only a subset of the vertices $V[l]$ are retained. We want the decimation to be performed in parallel on $G[l]$.

We will define a decimation process that is dependent on the image data. We assume that every cell c_i (a vertex of $G[l]$) carries a value g_i characterizing its region of the image—for example, the average gray level of the region. Without loss of generality we can assume that g_i is a scalar value; the treatment of feature vectors is identical. From now on the explicit indication of the level l will be dropped to simplify the notation.

Let cell c_0 on level l have r neighbors on level l, i.e., let its degree as a vertex of $G[l]$ be r. (Note that for the moment c_0 is not necessarily a survivor and the set of its neighbors has no relation with the connected subset allocated to a parent.) We examine every neighbor c_i, $i = 1, \cdots, r$ of c_0 and decide whether or not it belongs to the same "class" as c_0. This decision can depend in any desired way on the values $g_i, i = 0, \cdots, r$. We associate a binary number λ_i, $i = 0, \cdots, r$ with each neighbor, where $\lambda_i = 1$ if c_i belongs to the same class as c_0, and $\lambda_i = 0$ otherwise; note that $\lambda_0 = 1$.

The decimation algorithm employs three variables for every cell: two binary state variables p and q, and a random variable uniformly distributed between $[0, 1]$ with outcomes x. The survivors are chosen by an iterative local process. Let $k = 0, 1, \cdots$ be the iteration index. Initially all $p_i(0) = 0$. A cell survives if at the end of the algorithm its $p_0(k)$ state variable has the value 1.

Every iteration has two steps. First $q_0(k)$ is updated based on the states $p_i(k-1)$ of neighboring cells in the same class:

$$\begin{aligned} q_0(k) &= 1 \qquad \text{if } \lambda_i p_i(k-1) = 0 \quad \forall\ i = 0, \cdots, r \\ q_0(k) &= 0 \qquad \text{otherwise.} \end{aligned} \tag{4}$$

In other words, $q_0(k)$ becomes 1 if and only if there is no survivor among the cells belonging to the same class in the neighborhood of c_0. Note that the neighborhood includes the cell itself. The initial conditions always yield $q_0(1) = 1$. Then $p_0(k)$ is computed on the updated values of $q_i(k)$:

$$\begin{aligned} p_0(k) &= 1 \qquad \text{if } q_0(k)x_0(k) = \max_{i=0,\cdots,r} (\lambda_i q_i(k) x_i(k)) > 0 \\ p_0(k) &= p_0(k-1) \quad \text{otherwise.} \end{aligned} \tag{5}$$

To become a survivor the outcome of the random variable x drawn by the cell must be the local maximum among the outcomes drawn by the neighbors in the same class. Note that only those neighbors are taken into account which do not already have a survivor adjacent to them $(q_i(k) = 1)$. This condition extends the region of influence of a cell beyond its immediate neighborhood and yields faster convergence of the algorithm. The local maximum property assures that (2) is always satisfied. The state of a survivor is not reversible. Once a cell is labeled $p_0(k-1) = 1$, at subsequent iterations the product $q_0(k)x_0(k)$ (5) is always 0 by the definition of $q_0(k)$ (4). Thus in (5) the second condition, preserving the current state is used. It can be shown [22], [23] that after a finite number of iterations (at most five, in the experiments reported there) the algorithm reaches a final global configuration in which the survivors satisfy (1) as well.

The algorithm is entirely local, every cell computing its states based only on the states of its immediate neighbors. Except at the highest levels of the hierarchy, where due to the small number of vertices artifacts may occur, the decimation ratio between two consecutive levels exceeds four. This lower bound results from the fact that two adjacent cells never survive. On the random graph structure of higher pyramid levels the average degree of a vertex is around 6 [23]. To satisfy the nonadjacency condition (2) the number of vertices must be reduced by about the same order relative to the previous level.

By employing the algorithm an irregular sampling hierarchy can be built in parallel in log(class_size) steps. (The distinction between class_size and image_size becomes clear in the next section.) The stochastic decimation is performed independently within classes. In the next two sections we describe two applications of the process, one to connected component analysis of labeled images, the other to segmentation of gray level images.

III. Connected Component Analysis of Labeled Images

In a labeled image the pixels are classified into a small number of classes distinguished by different labels. A connected component is a maximal set of connected pixels sharing the same label. For simplicity we will restrict ourselves to the case where there are only two labels, i.e., to the case of a binary image, but images with multiple labels can be handled in essentially the same fashion.

Sequential algorithms for analyzing the connected components in a binary image usually employ a row-by-row scan [30]. An alternative approach makes use of the quadtree representation of the image [34]. In this section we apply the techniques described in Section II to obtain in log(class_size) steps a description of the connected components in a binary image. The description takes the form of a graph whose vertices represent the components and whose edges represent the adjacency relations among the components.

The fact that the pixels are labeled makes classification of the neighbors of a cell immediate. Let the label of cell c_i be g_i. In the binary case the label can have only two values. Thus in the neighborhood of cell c_0 we have for $i = 0, \cdots, r$ the class membership variables

$$\begin{aligned} \lambda_i &= 1 \qquad \text{if } g_i = g_0 \\ \lambda_i &= 0 \qquad \text{if } g_i \ne g_0. \end{aligned} \tag{6}$$

Note that (6) is symmetrical; c_0 gets the same value of λ in the neighborhood of c_i as c_i gets in c_0's neighborhood. Since the definition of λ_i is symmetrical it can be regarded as the weight

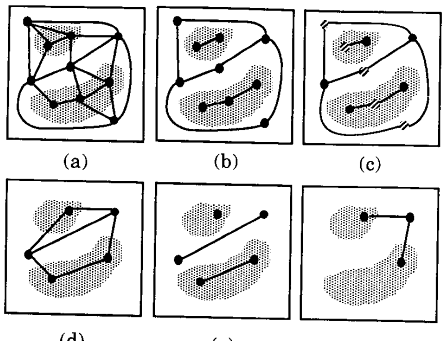


Fig. 2. Example of extracting connected components from a binary image. (a) $G[l]$. (b) $G'[l]$. (c) Survivor cells (filled circles) and the allocation of children (hashed circles) at level l. (d) $G[l+1]$. (e) $G'[l+1]$. (f) $G[m]$ for $m = l + 2$, the root level.

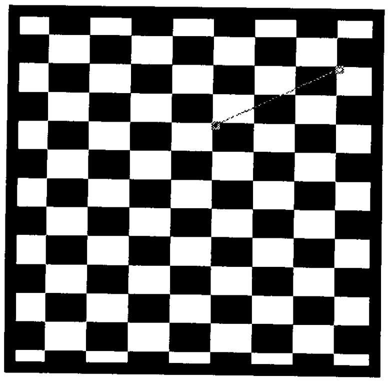


Fig. 3. The *checkerboard* image with the adjacency graph of the root level superposed.

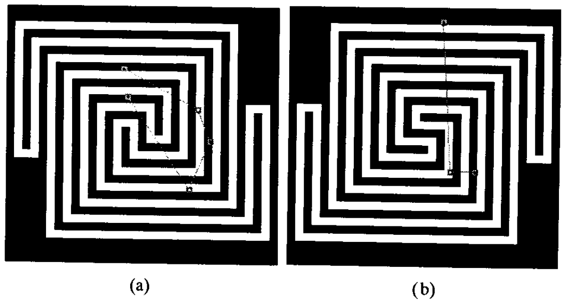


Fig. 4. The connectedness puzzle of Minsky and Papert. (a) Two white and three black bands. (b) One white and two black bands. The adjacency graph of the root level is superposed.

of the edge (c_0, c_i). The case $\lambda = 0$ is equivalent to removing the edge from $E[l]$. Let $E'[l]$ be the set of edges having $\lambda = 1$, and let $G'[l] = (V[l], E'\{l\})$. The connected components in the labeled image are represented by connected components in the graph $G'[l]$, for all $l \geq 0$.

The subgraphs of G' are processed independently, each subgraph being recursively contracted into one vertex, the *root* of the connected component. The contraction process is based on the technique described in the previous section: first the survivor vertices are designated and then the nonsurvivor vertices are locally allocated to survivors. If a nonsurvivor has more than one survivor neighbor it chooses the one carrying the largest outcome of the random variable x from the last iteration of the decimation process. Because the neighbors are neighbors in G', the survivors can only have children belonging to their own class. Thus from each connected component of the input image a pyramidal hierarchy of irregular tessellations is built in O(log(component_size)) steps.

The different hierarchies may have different heights, but in log[max(component_size)] steps the entire image is reduced to roots. This situation is detected at the level m when $E'[m]$ becomes empty. Evidently component_size can differ from image_size. For example, a connected linear pattern passing through every second row of the image has length $N(N+1)/2$ pixels. Since the hierarchy is built over the pattern the number of levels depends on its intrinsic diameter.

At each level, the graph $G[l]$ includes edges between cells that arise from different labels; it preserves the spatial relations among the connected components. At the root level, $G[m]$ is the adjacency graph of the original labeled image; it has one vertex for each connected component and its edges represent the adjacencies between these components.

Fig. 2.(a) shows an example of a graph $G[l]$ superposed on the binary image from which it was derived. The induced graph $G'[l]$ is shown in Fig. 2(b). Note that in $G'[l]$ each connected component corresponds to a connected subgraph. The cells surviving level l and the allocation of the nonsurvivors are shown in Fig. 2(c). The graph $G[l+1]$ of the next level is shown in Fig. 2(d) and the corresponding graph $G'[l+1]$ in Fig. 2(e). Level $l+2$ is the root level and its graph $G[m]$ is shown in Fig. 2(f). It correctly represents the adjacency relations among the three connected components of the image: the background and the two blobs.

In Fig. 3 a *checkerboard* image and the adjacency graph of its root level are shown. The *checkerboard* is a "worst-case" image, the two connected components (both defined by the relation of eight-connectedness) being distributed across the entire input. Cibulskis and Dyer [10] employed a regular pyramid structure to segment this image. In their results the "white" component was allocated to one root at the apex, but the representation of the black squares had to be spread over several levels. The size of the image is 64 × 64 and the two roots were obtained at the eight level of the hierarchy. Recall that the height of the hierarchy depends on component_size. Since random processes are involved in the construction of the irregular tessellations the location of the roots depends on the outcomes of local processes. Nevertheless, the same root level adjacency graph is always obtained at the top of the hierarchy.

The famous connectedness puzzle of Minsky and Papert [27, Fig. 5.1] can be solved by our technique in O(log(component_size)) steps. The pattern in Fig. 4(a) contains three black and two white bands, while in the pattern in Fig. 4(b) the two white bands are connected, leaving only two black bands. The adjacency graphs obtained at the root level clearly show the different topologies.

The irregular tessellations that arise in the hierarchies defined by the connected components do not convey meaningful representations at intermediate levels. Let us define the *receptive field* of a cell on level l as the set of all the pixels at level 0 (input) associated with it. This field is always a connected set

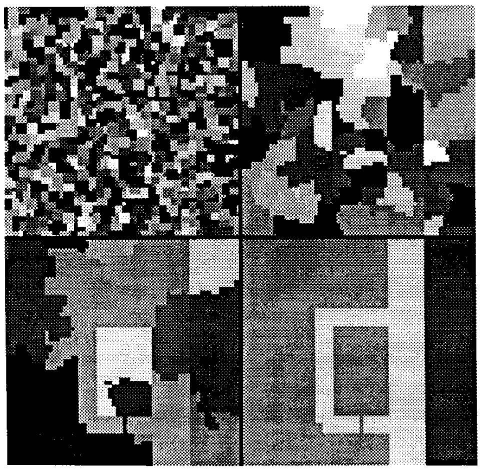


Fig. 5. Receptive fields of a hierarchy derived from a simple binary image. The fields are randomly colored. Top-left: First level. Top-right: Third level. Bottom-left: Fourth level. Bottom-right: Seventh (root) level. The fields correspond to the connected components.

and the image is the disjoint union of the fields. Also, each field is a subset of a connected component of the image. In Fig. 5 the receptive fields of four levels of a hierarchy derived from a simple image are shown. The different fields are randomly colored to emphasize their shapes. At intermediate levels the shapes of the fields are arbitrary, since they depend on the outcomes of random variables. At the root level each field is a complete connected component.

Our multiresolution representation consists of several independent hierarchies, each built independently over a connected component. The shape, size, position, and orientation of the connected component have no influence on the final result. The individual hierarchies can be used for the fast recovery of geometrical properties such as area or perimeter [32]. It should be mentioned that Miller and Stout [26] also proposed a data structure in which a separate "essential" regular pyramid is built over every object.

All the discussion in this section was restricted to binary images. If more than two labels are used the discussion is essentially identical. In particular, our method can be used to label the connected components of constant gray level in an arbitrary digital image. In the next section we study the less well-defined problem of segmenting a gray level image into "natural" regions.

IV. Segmentation of Gray Level Images

In gray level images the difference between the values of two adjacent pixels is bounded below only by the size of the quantization step. In our technique, to build the hierarchies the pixels in a neighborhood must be assigned to classes. The class membership induces the graph on which the stochastic decimation takes place. For labeled images the classes correspond to the labels and the hierarchy always converges to the same final representation: the adjacency graph of the connected components defined by the labels. For gray level images it is no longer obvious how to define the classes (unless our goal is to segment the image into connected components of constant gray level). In the first part of this section we discuss this problem.

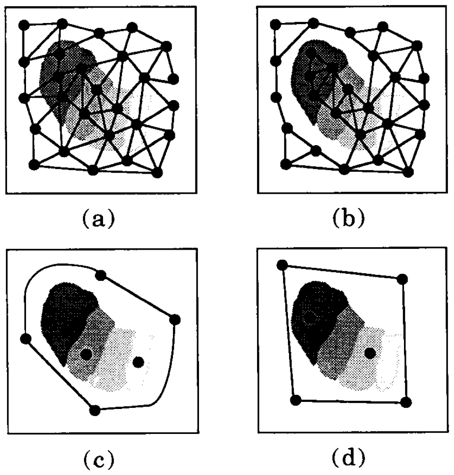


Fig. 6. Hierarchy derived from a gray level image. (a) $G[l_1]$. (b) $G'[l_1]$. (c) $G'[l_2]$, $l_1 > l_2$. (d) $G'[l_2]$ obtained for a different hierarchy structure.

The simplest approach is to define class membership by thresholding the gray level differences between the center cell c_0 and its neighbors c_i, $i = 1, \cdots, r$. The class membership variables λ_i are thus defined by

$$\begin{aligned} \lambda_i &= 1 \quad \text{if } \delta_i = |g_i - g_0| \leq T \\ \lambda_i &= 0 \quad \text{if } \delta_i = |g_i - g_0| > T. \end{aligned} \tag{7}$$

As in the labeled case, (7) based on an absolute threshold T is symmetrical. This symmetry, however, can create artifacts when we attempt to segment gray level images, as we show in the next example.

Fig. 6 shows an object having four gray levels on a white background. The graph $G[l_1]$ of an intermediate level is shown superposed on the image in Fig. 6(a). Let the differences between the gray levels be less than T and let the two lighter gray levels be within T of the background. The resulting graph $G'[l_1]$ is shown in Fig. 6(b). Note the edges connecting regions that have different colors. The stochastic decimation algorithm selects survivor cells and the nonsurvivors are allocated to their most similar surviving neighbors. The survivors (parents) compute new gray level values based on their children. After a few more levels of the hierarchy we might arrive at the graph $G'[l_2]$, $l_2 > l_1$, shown in Fig. 6(c). The difference between the gray levels of the two cells located in colored regions now exceeds T and in $G'[l_2]$ these regions are no longer connected. If a different set of outcomes of the random variables had been employed in the stochastic decimation process, a different set of surviving vertices might be obtained, and the new parents might have different gray level values, yielding a new graph at level l_2 [Fig. 6(d)]. We conclude that using a symmetric class membership criterion for gray level image segmentation strongly influences the structure of the hierarchy and therefore the final representation of the image.

Our next example, a *ramp* image, shows the severity of the resulting artifacts. In Fig. 7 (top-left) the image of a ramp going from level 0 (black) to level 255 (white) is shown. The difference between adjacent rows of pixels is either four or five gray levels, depending on the quantization error. The pixel values are the same along each row. The receptive fields of the root level obtained for T = 33 are shown in Fig. 7 (top-right). The color of a region is the gray level value computed by its

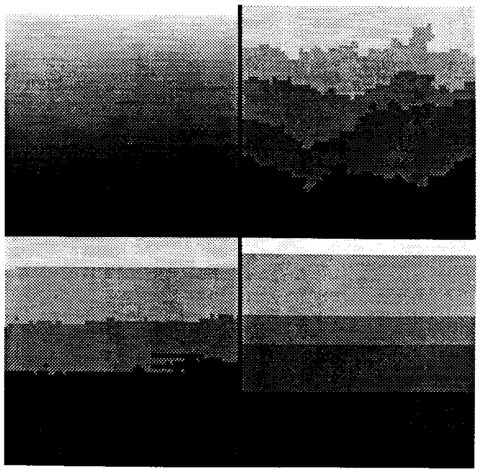


Fig. 7. The influence of stochastic decimation on the receptive fields of the root level derived from the *ramp* image. Top-left: Original image. Top-right: Using a symmetric class membership criterion. Bottom-left: Using a symmetric class membership criterion and reallocation of children. Bottom-right: Using a nonsymmetric membership criterion.

root, i.e., its average gray level in the original image. These receptive fields define a possible segmentation of the image, but the segmentation is not esthetically satisfactory; the different local configurations generated by the stochastic decimation yield ragged boundaries between the region. When at each level the children can be reallocated to adjacent parents having similar gray levels, the shapes of the segmented regions improve (Fig. 7, bottom-left). The procedure is essentially the same as the linking processes employed in traditional pyramid structures (e.g., [13], [16]). The receptive fields, however, may now be fragmented, and connectivity may not preserved, because of inconsistent exchanges of children.

To achieve satisfactory results, a nonsymmetric class membership criterion must be used. We now describe such a criterion. Let cell c_0 have r neighbors. In this neighborhood we will define a local threshold $S[c_0]$ such that $0 \leq S[c_0] \leq T$. Methods for computing $S[c_0]$ will be discussed later in this section. The definition of the class membership variables λ_i then becomes

$$\begin{aligned} \lambda_i = 1 \qquad & \text{if } \delta_i = |g_i - g_0| \leq S[c_0] \\ \lambda_i = 0 \qquad & \text{if } \delta_i = |g_i - g_0| > S[c_0] \end{aligned} \tag{8}$$

for $i = 0, 1, \cdots, r$. If the number of neighbors having $\lambda_i = 1$ is s and the number of neighbors for which $|g_i - g_0| \leq T$ is t, then $0 \leq s \leq t \leq r$. The threshold $S[c_0]$ used in (8) is specific to the neighborhood of cell c_0 and therefore the criterion is not symmetric: in general,

$$|g_i - g_0| \leq S[c_o] \tag{9}$$

does not imply

$$|g_0 - g_i| \leq S[c_i] \tag{10}$$

since the two thresholds are computed based on neighborhoods that only partially overlap. As a consequence of the nonsymmetry the graphs $G'[l]$ become directed. An arc from c_i to c_j, meaning that in the neighborhood of c_i the two cells belong to the same class, may not be reciprocated by an arc from c_j to c_i.

The receptive fields of the ramp image's root level for a hierarchy built with nonsymmetric class memberships (as described below) are shown in Fig. 7 (bottom-right). The same absolute threshold $T = 33$ was used. The boundaries between regions are now correctly delineated along the rows of the image. The neighborhood-dependent local threshold $S[c_0]$ ensures that every cell connects first to its neighbors that have the most similar gray level. Thus the individual rows in the image are reduced to single cells *before* two cells belonging to adjacent rows can become neighbors in the graph $G'[l]$. The effect of the random processes used in building the hierarchy is reflected by the different widths of the regions. However, this artifact can also be eliminated as we shall see at the end of the section.

We now discuss how the value of the local threshold $S[c_0]$ is computed. Note that the extreme case $S[c_0] = 0$ corresponds to extracting connected components of constant gray level. Several approaches are available to determine the $S[c_0]$ value that best dichotomizes the neighborhood into two classes. A wide class of thresholding methods (e.g., [33]) can be considered. Another approach is to use the neighbor dichotomization techniques employed in nonlinear image smoothing techniques (e.g., [15]). Since the neighborhoods are defined on the graphs, the spatial relations among the neighbors will not be used; we will employ only gray level information in computing $S[c_0]$.

Let $\delta_{[i]}$, $i = 0, 1, \cdots, r$ be the ordered sequence of absolute gray level differences $\delta_i = |g_i - g_0|$. Thus

$$\delta_{[0]} = 0 \leq \delta_{[1]} \leq \cdots \delta_{[s]} \leq S[c_0] < \cdots \delta_{[t]} \leq T < \cdots \delta_{[r]}. \tag{11}$$

The simplest way of defining $S[c_0]$ is the *most similar neighbor* method. In this case $S[c_0] = \delta_{[1]}$ and $s = \max \arg(\delta_{[i]} = \delta_{[1]})$. This is the method that we used for the *ramp* image (Fig. 7, bottom-right). In gray level images of real scenes with less regular structure this method could yield "tall" hierarchies, because the neighborhood sizes in the $G'[l]$ graphs are small, and therefore the decimation ratio between consecutive levels is too low. Generalizations of this method such as *k most similar neighbors* [12] $S[c_0] = \delta_{[k]}$, or *fixed fraction of good neighbors,* $S[c_0] = \alpha . \delta_{[t]}$, have the disadvantage that the constants k or α must be defined arbitrarily.

We have obtained good results using the *maximum averaged contrast* method in which $S[c_0]$ is set by detecting the most significant step in the sequence of $\delta_{[i]}$. For all the t neighbors within the absolute threshold gray level difference we compute

$$U_j = \frac{\sum_{i=1}^{j} \delta_{[i]}}{j}; \quad V_j = \frac{\sum_{i=j+1}^{t} \delta_{[i]}}{t - j} \qquad 1 \leq j \leq t - 1 \tag{12}$$

Let $u = \min \arg(\max_j (V_j - U_j))$ and $s = \max \arg(\delta_{[i]} = \delta_{[u]})$. The threshold $S[c_0] = \delta_{[s]}$ is the first occurrence of the maximum averaged contrast. For example, the sequence $\delta_{[1]} = \delta_{[2]} = \delta_{[3]} = 0, \delta_{[4]} = \delta_{[5]} = \delta_{[6]} = 1$ yields $S[c_0] = \delta_{[3]} = 0$. On the other hand, the sequence $\delta_{[1]} = 0, \delta_{[2]} = 1, \delta_{[3]} = 2, \delta_{[4]} = 3, \delta_{[5]} = 5, \delta_{[6]} = 5$ yields $S[c_0] = \delta_{[4]} = 3$.

The gray level value of a parent $g_0[l+1]$ is computed as the weighted average of its children's gray levels $g_i[l]$:

$$g_0[l+1] = \frac{\sum_{i=0}^{z} g_i[l] A_i[l]}{\sum_{i=0}^{z} A_i[l]} \tag{13}$$

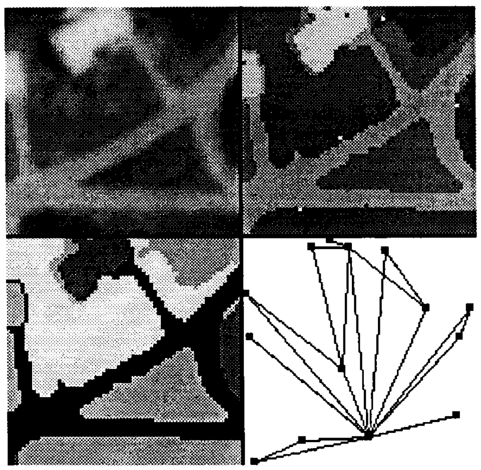


Fig. 8. Example of segmentation of the *aerial* image. All the images are magnified to 128 × 128. Top-left: Original image. Top-right: Receptive fields of the root level with the root locations superposed. The regions have the root's gray level; the color of the roots is for illustration only. Bottom-left: The randomly colored receptive fields. Bottom-right: Adjacency graph of the root level.

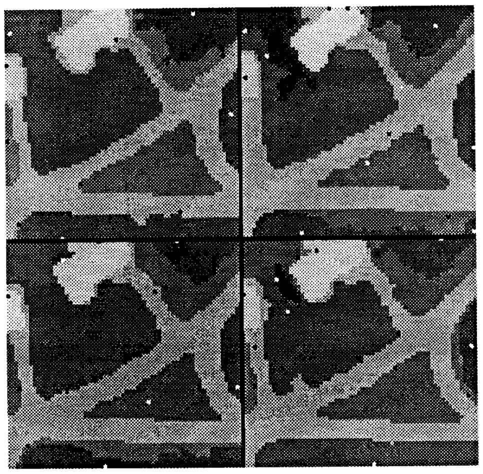


Fig. 9 The influence of hierarchy structure on the segmentation of the *aerial* image. Root locations superposed. Top-left: n = 10, 11 roots. Top-right: n = 8, 14 roots. Bottom-left: n = 10, 12 roots. Bottom-right: n = 10, 14 roots.

where to bias the segmentation toward larger regions, the receptive field area $A_i[l]$ (the number of pixels in the region) is used as the weight; z is the number of children.

To see how this method works on a real image, in Fig. 8 (top-left) a 64 × 64 *aerial* image magnified to 128 × 128 is shown. The receptive fields for the root level and the locations of the roots are shown at top-right. Since the gray levels of the roots are used to color the regions, adjacent fields may appear fused. A random coloring is shown in Fig. 8 (bottom-left). The root level's adjacency graph, describing the adjacencies among the regions, is shown at bottom-right. In this example the root level is at m = 8. We have found that hierarchies built for this image with different random variable outcomes can have m as high as 10, significantly larger than n = 6, the number of levels in a regular pyramid structure for a 64 × 64 image. The use of stochastic decimation adapted to the local structure of the image slows down the convergence of the process. As mentioned before, the most similar neighbor method yields the slowest convergence; in one hierarchy built using this method we obtained m = 15. On the other hand, using an absolute threshold (T = 33) usually generates hierarchies with m = n = 6 because the neighborhood sizes used in the decimation are much larger.

Different outcomes of the random variables cause changes in the hierarchy structure. For labeled pictures the final result of the analysis, i.e., the connected component description at the root level, is always the same; but this is not the case for segmentation of gray level images. By changing the set of survivor cells the values attributed to these cells may also change slightly, yielding changes in the graphs on subsequent levels. Four different segmentation results for the *aerial* image are shown in Fig. 9. In each example different random variable outcomes were employed, and the absolute threshold was always T = 33. All the figures but the top-right have m = 10, while the top-right has m = 8. The two-level height difference is mainly caused by the relative inefficiency of the stochastic decimation process when the similarity graph has only a small number of vertices. The number of roots is 11 for the top-left image, 12 for the bottom-left, and 14 for the two images on the right. As expected, regions with sharp boundaries in the input image (Fig. 8, top-left) achieve very similar representations. The effect of the stochastic processes is more significant in the segmentation of smoothly changing regions.

Our method provides the adjacency graph of the segmented image and a separate hierarchy ("pyramid") for every region. These tools can be used by model driven vision modules to analyze properties of the regions and correct the segmentation in a logarithmic number of processing steps.

The value of the absolute threshold T influences the result of the segmentation by defining the "good" neighbors of a cell that are taken into account in defining class membership (8). Modifying T thus changes the structure of the hierarchies. In Fig. 10 a *medical* image and its segmentation for three different values of T are shown. We chose this image since many of the features are less well delineated than those in the *aerial* image. The height of the root level and the number of roots in each case are given in the legend of the figure. The larger T, the smaller the number of roots since the fusion of adjacent classes is more probable. The correlation between T and the height of the root level m is less immediate. At higher levels the decimation process may slow down because of the smallness of the graphs. The effect depends on the structure of the hierarchy, and therefore also on T. Note that the *medical* image is noisy (Fig. 10, top-left) but only a few noisy pixels remain at the root level. These single pixel roots are easy to identify and remove from the segmented image.

Another example is shown in Fig. 11. The *house* image (left) was analyzed with T = 26. The height of the root level in this example is m = 12 and 100 roots were extracted (right). Note the accurate delineation of shadows and of the small feature over the garage door. Segmentation of such complex images is sensitive to the value of T. When we used T = 33 the sky and the roof fused in some of the hierarchies. However, once the image representation is obtained, individual analysis of the segmented objects can be performed to correct any undesired fusion by lowering the threshold.

In gray level image analysis, the problem of root detection also requires careful treatment. In labeled pictures a root (an

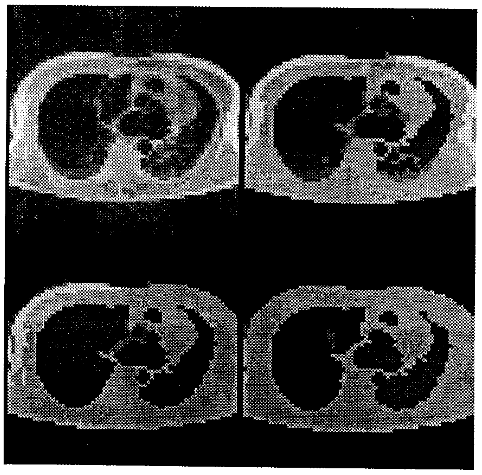


Fig. 10. The influence of the absolute threshold on the segmentation of the *medical* image magnified from 64 × 64 to 128 × 128. Top-left: Original image. Top-right: $T = 29$, $m = 13$, 67 roots. Bottom-left: $T = 39$, $m = 12$, 30 roots. Bottom-right: $T = 48$, $m = 15$, 13 roots.

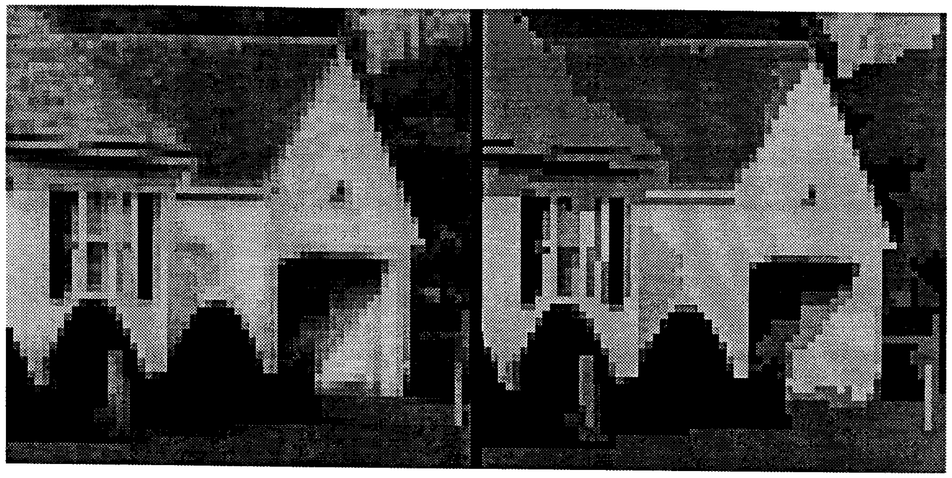


Fig. 11. The *house* image magnified from 64 × 64 to 256 × 256. Left: Original image. Right: Segmented image, $T = 26$, $m = 12$, 100 roots.

isolated vertex in the graph $G'[l]$) remains a root at higher levels, while other (larger) connected components of the image are being extracted. In gray level images this is not true. The value of an already extracted root may become within threshold of some neighbor's value at a higher level. The root cell then becomes connected in the graph $G'[l]$ of that level and must be taken into account in the stochastic decimation process. Thus a root may disappear at subsequent levels, its receptive field being fused into a larger region. Nevertheless, we found this approach more desirable than the following alternatives:

- An already detected root always survives decimation, but vertices of the upper levels' graphs may become its children. In this case at the beginning of the decimation process the root already has $p = 1$, i.e., it is already a survivor. The nonuniform initial condition for decimation, however, could result in a connected object being represented by several roots on the root level.
- An already detected root is never taken into consideration in later $G'[l]$ graphs. This approach tends to retain too many roots and oversegments the image.

Additional control over the result of segmentation is gained if changes in the local thresholds computed by a cell on successive levels are taken into account. Let $S_l[c_0]$ be the local threshold of a cell surviving the decimation of level l. Assume that on level $l + 1$ the cell computes the new threshold $S_{l+1}[c_0]$. A more detailed segmentation is obtained if we declare the cell to be a root when

$$\frac{|S_{l+1}[c_0] - S_l[c_0]|}{S_{l+1}[c_0]} > \Delta \tag{14}$$

where Δ is a constant. For example, by using a small Δ in the *ramp* image (Fig. 7, bottom-right) we can stop the fusion of adjacent rows. At the root level the adjacency graph becomes linear, with every vertex now representing an entire row of the image. We can now apply the stochastic decimation algorithm to this one-dimensional structure to allocate to every vertex its address in the corresponding linked list. This operation is also achieved in O(list_size) steps [22]. Thus the adjacency graph can be contracted into a graph in which the vertices correspond to collections of rows; the new graph defines the segmentation of the ramp into constant width bands.

V. Discussion

In this paper we have presented an image analysis technique in which a stochastic decimation algorithm constructs a tessellation hierarchy that reflects the structure of the image. For labeled images the final tessellation is into connected components, and is unique. For gray level images the tessellation is not unique, but it constitutes a reasonable segmentation of the image.

If our methods can be implemented on suitable parallel hardware, every root can recover information about its region in a logarithmic number of processing steps, and the adjacency graph can become the foundation for a relational model of the scene. On appropriate hardware our technique should be useful in real-time analysis of the visual environment. We intend to implement our methods on the Connection Machine where the architecture allows communication among any two processors in a few steps. This is essential since on a graph any two processors can become neighbors.

Any definition of the classes can be used when building the hierarchy. For example, the definition of the classes can take into account the properties of the corrupting noise if they are available, or positional information derived from the previous level. In the latter case, however, the data driven component of the method is weakened. The decimation process can be modified to be biased toward cells with high informational value. Jolion and Montanvert [17] have proposed an adaptive pyramid in which cells belonging to the most homogeneous regions have priority to become survivors. Such an approach, however, is not useful for labeled images in which many cells carry identical descriptions.

The evolution of the local connections within the hierarchy, driven by both the image data and the stochastic processes, is itself of interest, and might serve as a neural model for early visual perception. Appropriate class membership criteria might transform the hierarchy in a connectionist model for extraction of perceptual invariants [3].

The graph contraction used to build the hierarchy satisfies two constraints: 1) a removed vertex always has a retained neighbor; 2) two adjacent vertices cannot both survive. In graph theory, finding a surviving subset of vertices is known as the maximal independent set problem (e.g., [21]). The stochastic decimation process that we employed provides such a subset and therefore solves the problem. Our algorithm is different from other solutions proposed in the literature. In these methods vertices are first chosen at random and then some of them are discarded in order to have the constraints satisfied. Such trial-and-error approaches are considerably slower than our algorithm, which has the constraints embedded within the selection process. Simulations have shown a fourfold speed-up for the stochastic decimation procedure relative to other algorithms.

Acknowledgment

We would like to thank W. Kropatsch for challenging us with the checkerboard example, S. Banerjee for calling our attention to the similarity between the decimation process and the maximal independent set problem, and D. Y. Kim for suggesting the local threshold computation.

References

[1] E.H. Adelson, E. Simoncelli, and R. Hingorani, "Orthogonal pyramid transforms for image coding," in *Visual Communications and Image Processing II, SPIE Proc.*, vol. 845, pp. 50–58, 1987.
[2] N. Ahuja, "On approaches to polygonal decomposition for hierarchical image representation," *Comput. Graphics Image Processing,* vol. 24, pp. 200–214, 1983.
[3] D.H. Ballard, "Cortical connections and parallel processing: Structure and function," *Behavioral Brain Sci.,* vol. 9, pp. 67–120, 1986.
[4] S. Baronti, A. Casini, F. Lotti, L. Favaro, and V. Roberto, "Variable pyramid structure for image segmentation," *Comput. Vision Graphics Image Processing,* vol. 49, pp. 346–356, 1990.
[5] M. Bister, "A critical view on pyramid segmentation algorithms," Inform. Retrieval and Interpretation Sci. Lab., Vrije Univ., Brussels, Belgium, Tech. Rep. IRIS-TR-0006, 1989.
[6] P.J. Burt, "Tree and pyramid structures for coding hexagonally sampling binary images," *Comput. Graphics Image Processing,* vol. 14, pp. 171–180, 1980.
[7] ——, "Fast filter transforms for image processing," *Comput. Graphics Image Processing,* vol. 16, pp. 20–51, 1981.
[8] V. Cantoni and S. Levialdi, Eds., *Pyramidal Systems for Computer Vision.* Berlin: Springer-Verlag, 1986.
[9] J.M. Chassery and A. Montanvert, "A segmentation method in a Voronoi diagram environment," *Proc. Sixth Scandinavian Conf. Image Processing,* Oulu, Finland, June 19–22, 1989, pp. 408–415.
[10] J. Cibulskis and C.R. Dyer, "Node linking strategies in pyramids for image segmentation," in *Multiresolution Image Processing and Analysis,* A. Rosenfeld, Ed. Berlin: Springer-Verlag, 1984, pp. 109–120.
[11] J.L. Crowley and R.M. Stern, "Fast computation of the difference of low-pass transform," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. PAMI-6, pp. 212–222, 1984.
[12] L.S. Davis and A. Rosenfeld, "Noise cleaning by iterated local averaging," *IEEE Trans Syst., Man, Cybern.,* vol. SMC-8, pp. 705–710, 1978.
[13] A.D. Gross and A. Rosenfeld, "Multiresolution object detection and delineation," *Comput. Vision Graphics Image Processing,* vol. 39, pp. 102–115, 1987.
[14] N.P. Hartman and S.L. Tanimoto, "A hexagonal pyramid data structure for image processing," *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-14, pp. 247–255, 1984.
[15] D. Harwood, M. Subbarao, H. Hakalahti, and L.S. Davis, "A new class of edge-preserving smoothing filters," *Pattern Recognition Lett.,* vol. 6, pp. 115–162, 1987.
[16] T.H. Hong and A. Rosenfeld, "Compact region extraction using, weighted pixel linking in a pyramid," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. PAMI-6, pp. 222–229, 1984.
[17] J.M. Jolion and A. Montanvert, "La pyramide adaptive: construction et utilisation pour l'analyse de scenes 2D," in *Proc. Seventh RFIA Conf.,* Paris, Nov. 1989, pp. 197–206.
[18] W.G. Kropatsch, "A pyramid that grows by powers of 2," *Pattern Recogni. Lett.,* vol. 3, pp. 315–322, 1985.
[19] W.G. Kropatsch, "Curve representation in multiple resolutions," *Pattern Recognition Lett.,* vol. 6, pp. 179–184, 1987.
[20] M. Li, W.I. Grosky, and R. Jain, "Normalized quadtrees with respect to translations," *Comput. Graphics Image Processing,* vol. 20, pp. 72–81, 1982.
[21] M. Luby, "A simple parallel algorithm for the maximal independent set problem," in *Proc. Seventeenth Annu. ACM Symp. Theory of Computing,* 1985, pp. 1–10.
[22] P. Meer, "Stochastic image pyramids," *Comput. Vision Graphics Image Processing,* vol. 45, pp. 269–294, 1989.
[23] P. Meer and S. Connelly, "A fast parallel method for synthesis of random patterns," *Pattern Recognition,* vol. 22, pp. 189–204, 1989.
[24] P. Meer, E.S. Baugher, and A. Rosenfeld, "Frequency domain analysis and synthesis of image pyramid generating kernels," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. PAMI-9, pp. 512–522, 1987.
[25] R. Miller and Q.F. Stout, "Data movement techniques for the pyramid computer," *SIAM J. Comput.,* vol. 16, pp. 38–60, 1987.

[26] —— "Simulating essential pyramids," in *Proc. IEEE Comput. Soc. Conf. Computer Vision and Pattern Recognition,* Ann Arbor, MI, June 5–9, 1988, pp. 912–917.

[27] M. L. Minsky and S. A. Papert, *Perceptrons,* expanded ed. Cambridge, MA: MIT Press, 1988.

[28] S. Peleg, O. Federbusch, and R. Hummel, "Custom-made pyramids," in *Parallel Computer Vision,* L. Uhr, Ed. New York: Academic, 1987, pp. 125–146.

[29] H. Rom and S. Peleg, "Image representation using Voronoi tessellation: Adaptive and secure," in *Proc. IEEE Comput. Soc. Conf. Computer Vision and Pattern Recognition,* Ann Arbor, MI, June 5–9, 1988, pp. 282–285.

[30] C. Ronse and P. A. Devijver, *Connected Components in Binary Images: The Detection Problem.* Letchworth, England: Research Studies Press, 1984.

[31] A. Rosenfeld, Ed., *Multiresolution Image Processing and Analysis.* Berlin: Springer-Verlag, 1984.

[32] A. Rosenfeld, "Recognizing unexpected objects: A proposed approach," *Int. J. Pattern Recognition Artificial Intell.,* vol. 1, pp. 71–84, 1987.

[33] P. K. Sahoo, S. Soltani, and A. K. C. Wong, "A survey of thresholding techniques," *Comput. Vision Graphics Image Processing,* vol. 41, pp. 233–260, 1988.

[34] H. Samet, "The quadtree and related hierarchical data structures," *Comput. Surveys,* vol. 16, pp. 187–260, 1984.

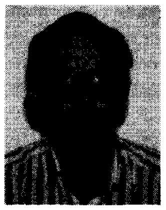
[35] C. A. Shaffer, "A formula for computing the number of quadtree node fragments created by a shift," *Pattern Recognition Lett.,* vol. 7, pp. 45–49, 1988.

[36] S. Tanimoto, "Pictorial feature distortion in a pyramid," *Comput. Graphics Image Processing,* vol. 5, pp. 333–352, 1976.

[37] L. Uhr, Ed., *Parallel Computer Vision.* New York: Academic, 1987.

Annick Montanvert (S'84–M'87) was born in Grenoble, France, on November 14, 1960. She received the Diplôme d'Ingénieur from the Ecole Nationale Supérieure d'Ingénieurs en Informatique de Grenoble in 1984, and the Ph.D. degree in computer science from the University of Grenoble in 1987.

She is currently Maitre de Conférences in Computer Science at the University of Grenoble II and works on Image Analysis in the Laboratory TIM3 of the IMAG group. Her research interests concern shape analysis using skeleton representations and image analysis with pyramidal approaches. In 1989 she spent six months at the Center for Automation Research, University of Maryland, as a postdoctoral fellow.



Peter Meer (S'84–M'86) was born in Oradea, Romania, on February 14, 1949. He received the Dipl. Engn. degree from the Bucharest Polytechnic Institute, Bucharest, Romania, in 1971, and the D.Sc. degree from the Technion, Israel Institute of Technology, Haifa, Israel, in 1986, both in electrical engineering.

From 1971 to 1979 he was with the Computer Research Institute, Cluj, Romania, working on R&D of digital hardware. From 1980 to 1986 he was with the Technion, developing computational models of human vision. Between 1986 and 1990 he was with the Center for Automation Research, University of Maryland, as an Assistant Research Scientist. In 1991 he joined the faculty of the Department of Electrical and Computer Engineering, Rutgers University, New Brunswick, NJ, as an Assistant Professor. His research interests include applications of robust statistical methods in computer vision and consensus based vision paradigm.

Azriel Rosenfeld (M'60–F'72) was born in New York City on February 19, 1931. He received rabbinic ordination in 1952, the Doctor of Hebrew Literature degree from Yeshiva University, New York, NY, in 1955, and the Ph.D. degree in mathematics from Columbia University, New York, NY, in 1957.

He is a tenured Research Professor and Director of the Center for Automation Research, a department-level unit of the University of Maryland in College Park. He also holds Affiliate Professorships in the Departments of Computer Science and Psychology and in the College of Engineering. He is a widely known researcher in the field of computer image analysis. He wrote the first textbook in the field (1969); was a founding editor of its first journal (1972); and was co-chairman of its first international conference (1987). He has published over 20 books and over 400 book chapters and journal articles, and has directed over 30 Ph.D. dissertations.

Dr. Rosenfeld won the IEEE Emanuel Piore Award in 1985; he was a founding Director of the Machine Vision Association of the Society of Manufacturing Engineers (1985–1988), and won its President's Award in 1987; he was a founding member of the IEEE Computer Society's Technical Committee on Pattern Analysis and Machine Intelligence (1965), served as its Chairman (1985–1987), and received the Society's Meritorious Service Award in 1986; he was a founding member of the Governing Board of the International Association for Pattern Recognition (1978–1985), served as its President (1980–1982), and won its first K. S. Fu Award in 1988; he is a Fellow of the Washington Academy of Sciences (1988), and won its Mathematics and Computer Science Award in 1988; he is a Corresponding Member of the National Academy of Engineering of Mexico (1982) and a Foreign Member of the Academy of Science of the German Democratic Republic (1988). He holds an honorary Doctor of Technology degree from Linköping University, Sweden (1980), and is a certified Manufacturing Engineer (1988).