Technical Report CS79010-R

TEXTURE REGION GROWING BASED UPON A STRUCTURAL MODEL OF TEXTURE

by

Roger W. Ehrich

Peichung F. Lai*

Virginia Polytechnic Institute

and State University

Blacksburg

November, 1979

*P.F. Lai is with the Department of Computer Science, University of Alabama, University, Alabama

Abstract

In this paper the problem of texture boundary formation is approached by a region growing technique that is based upon a structural model of texture. Region growing is based upon similarities among texture elements and upon the spatial proximities of these elements. The region grower itself is a global algorithm that makes use of a minimal spanning tree of what is called the associated texture graph of a texture. A primary advantage of the method is that the texture regions are self-organizing in the sense that no artificial windows need be superimposed over the source textures. While the region described in this paper uses extrema-based texture elements, the application to other types of elements is straightforward.

This work was supported by the National Science Foundation under grant ENG-7712105.

1. Introduction

The problem of texture analysis is important to investigators in the various fields of image processing because almost all natural scenes are composed of regions characterized by texture. Various applications such as recognition of biological specimens, blood cell diagnosis, and multispectral imagery classification make use of texture processing techniques as tools in the analysis.

In spite of the importance of texture in the area of image processing, there is no precise scientific definition of texture. In Webster's dictionary, texture is defined as "the character of a fabric determined by the arrangement, size, shape, etc., of its threads, or the arrangement of constituent parts of anything." From the point of view of image processing, if one views a picture as a two-dimensional spatial function, the picture intensity is the value of the function at a certain spatial location. Then a texture is basically a region in which there is substantial spatial variation in intensity that is due to a regular or statistical spatial distribution of texture elements. The primary characteristics of a texture are therefore variation and repetition. Texture also can be classified as being regular or statistical, depending upon whether the variations in texture elements and their interrelationships are described by algorithms or by probability density functions, respectively.

There are three main problems in texture studies. They are (1) texture discrimination, (2) texture description, and (3) texture boundary detection. These three problems are listed in order of increasing difficulty. Humans do well in perceiving texture for several important reasons. First, a very large number of textural features are extracted in parallel, and second, the visual system has an amazing ability to find organizations within a texture.

In a structural model of texture, the basic components such as line segments or circles from which a texture is constructed are called textural primitives. These may be organized into more complicated structures called texture elements which are replicated throughout the texture. A texture element is either a primitive or a recursive composition and nesting of texture elements. It is hypothesized that significant aspects of human texture perception can be simulated by extracting texture elements and their relationships, by determining textural features, and by detecting homogeneous organizations of texture elements.

Structural models of texture have been proposed by a number of investigators including Ehrich and Foith [Eh78], Lu and Fu [Lu78], and Zucker [Zu76]. Common to all of these models are the two main components of a structural model - the texture elements from which a texture is formed and the relationships among the texture elements that determine where the elements are to be placed and perhaps how they are to look.

Zucker distinguishes between ideal textures, which are represented by regular graphs with primitives at the nodes, and observable textures, which are obtained from ideal textures by stochastic noise and distortion processes. His model corresponds to a competence theory of texture description and has not been extended to a theory that can account for the performance of humans in perceiving textures. Lu and Fu assume that the relationships among texture elements can be specified by a tree grammar, and they have used a syntactic approach for texture synthesis and analysis. The spatial structure within constant sized windows of the texture image is expressed as a tree. The assignment of intensities to image points is determined by the rules of a stochastic tree grammar. Finally, the placement of windows is given by a higher level syntax in order to preserve the pictorial coherence between windows. Ehrich and Foith consider texture elements that are constructed from extrema-based primitives, and in their model the interrelationships among texture elements are determined directly from the topology of the picture function.

All three of the above models seem to suffer from uncertainties about the nature of the texture elements and from the difficult problem of determining which relationships among texture elements are important in texture analysis. For example, there are several possible interpretations of the simple texture shown in Figure 1. The first interpretation is that the texture consists of two superimposed textures, one of which consists of small circles with the other consisting of large circles. The second interpretation is that the left subregion consisting of large and small circles is a mixture of large and small circles that has its own physical origin and which is unrelated to the rest of the texture sample. The first interpretation

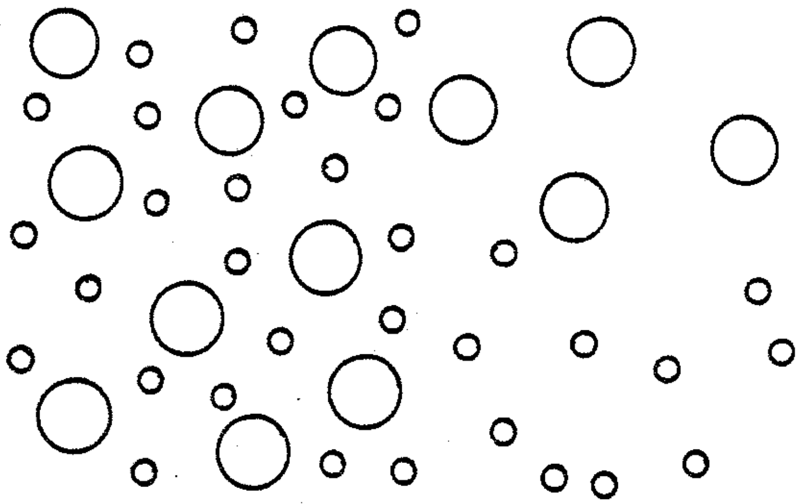


Figure 1 - Ambiguous texture.

leads to the identification of two overlapping regions of different texture, while the second interpretation leads to three disjoint regions of different texture. It is clear that scene context is required to resolve the ambiguity, and there are three mechanisms by means of which the two alternative hypotheses can be produced:

1) The texture analyzer can generate both hypotheses
2) The texture analyzer can produce the two region hypothesis, and the alternative interpretation is produced by the vision system by intersecting the overlapping regions, or
3) The texture analyzer can produce the three region hypothesis, and the alternative interpretation is produced by the vision system by extending the disjoint regions into one another on the basis of the similarities of the texture elements they contain.

Actually, the two hypotheses for Figure 1 are produced by complex interactions between the gestalt grouping principles of proximity and similarity. For the first interpretation, similarity dominates over proximity in the mixture region, since associations among texture elements are not formed by measuring spacing. In the second hypothesis, proximity dominates in the mixture region because the density of primitives is higher there than in adjacent regions. The interpretation would be biased toward the three region interpretation by proximity if the small circle density was lower in the mixture region. If some large circles in the mixture region also contained small circles, a third hypothesis that there were actually three distinct types of primitives would also have to be produced. It is not hard to see that both the number of suitable primitives and the number of homogeneous organizations that have to be considered might be very large. Therefore, it is not surprising that in much work on texture, very limiting assumptions have been made about the types of primitives and the spatial relationships to be considered.

Most of the techniques that have been used to locate texture boundaries seem to fall into two main categories. In the first category, a regular grid is superimposed over a scene, and it is assumed that within each cell is a homogeneous texture with measurable properties. Then one places boundary elements between adjacent cells whose textural properties are statistically different. The second category of techniques contains clustering techniques that group primitives together on the basis of similarity.

As is well known, the grid method suffers from the problem of having to select a cell size. If the cell size is too small there is a serious difficulty in making meaningful statistical decisions about the cell contents based upon a very small sample size. If the cell size is too large, the texture boundaries become inaccurate, and there is an increasing probability of finding several different texture types within a single cell. Generally, the cell sizes used tend to be small, and it is convenient to use individual picture elements as primitives. Relationships between primitives at various separation distances were determined by computing autocorrelation functions [Ka55] or Fourier transforms [Ba76]. One of the most successful techniques involved estimating the statistics of pairwise cooccurrences of primitives at fixed distances in particular directions by computing and analyzing cooccurrence matrices [Ha73, De73]. In addition to the problem of selecting an appropriate cell size, these methods suffered from the problem that except in the case of microtextures, primitives are usually composed of many picture elements, and the wrong kinds of statistical measurements were being made.

Central to clustering approaches were the histogramming techniques made popular by Tsuji [To73]. The basic idea was to compute local image properties, estimate their pdf's by histogramming them, separate the modes of the resulting histograms, and then determine the regions that, by hypothesis, were responsible for the histogram modes [Zu75].

One of the more useful local features is edge density [Ro70, Ro71]. First an edge image is obtained by applying a standard local gradient operator to the original image, and then each picture element is replaced by an average over its local neighborhood. Another useful local feature is the density of local extrema. Carlton and Mitchell [Ca77] use a hysteresis smoothing algorithm to detect only the local extrema which exceed a certain size. By using three threshold values they produced three intermediate binary images which showed the locations of the extrema at three levels. Then another three images are obtained from these by blurring each intermediate binary image. These and the blurred original were then used for segmentation by simple thresholding.

The histogramming approach has also been extended to include the measurement of cooccurrences of local properties [Dy79]. The main problem with histogramming is that regions are formed independently of the spatial relationships among members of the same histogram mode; consequently, elaborate heuristics are required to fill holes in regions and to eliminate noise points that form small, undesired regions.

2. Texture Region Growing

By now it may have occurred to the reader that if clustering approaches to region growing have difficulties because they do not take into account spatial relationships, then perhaps it might be possible to adapt some of the classical region growers to the problem of texture region growing. However, a region grower such as that of Muerle and Allen [Mu68] is restricted to use of spatially adjacent primitives. The more advanced region growers of Brice and Fennema [Br70] and Yakimovsky and Feldman [Ya73] make use of boundary information which is available only if the texture region boundaries happen to coincide with differences in average gray value.

The work presented in this paper attempts to address the problem of detecting textural boundaries by using global region growing techniques based upon a structural model of texture. A key aspect of the technique is the general way in which the spatial relationships among texture elements are used. In the proposed structural model, there are four main issues of concern; (1) texture elements, (2) attributes of texture elements, (3) relationships among elements, and (4) aggregation mechanisms.

First, texture primitives are extracted from the original picture. The original picture is transformed into a tree structure called a relational tree (or simply R-tree) which represents the hierarchical relationships among texture elements. Also, attributes associated with each texture element are computed and stored in the R-tree. Then texture region growing is done by merging or grouping textural elements together according to their global relationships. In the structural texture model, the first two issues concern the formulation of texture elements and the measurements of their associated attributes, while the other two issues are closely related to the region growing procedures.

2.1 Relational Trees

Ehrich and Foith [Eh76] describe a relational tree representation for one-dimensional intensity profiles which is briefly discussed here because it is the basis for much of the work described in this paper. A relational tree recursively partitions a profile into smaller and smaller segments based upon the values of minima in the profile. For example, the lowest valley c in Figure 2a is used for the initial

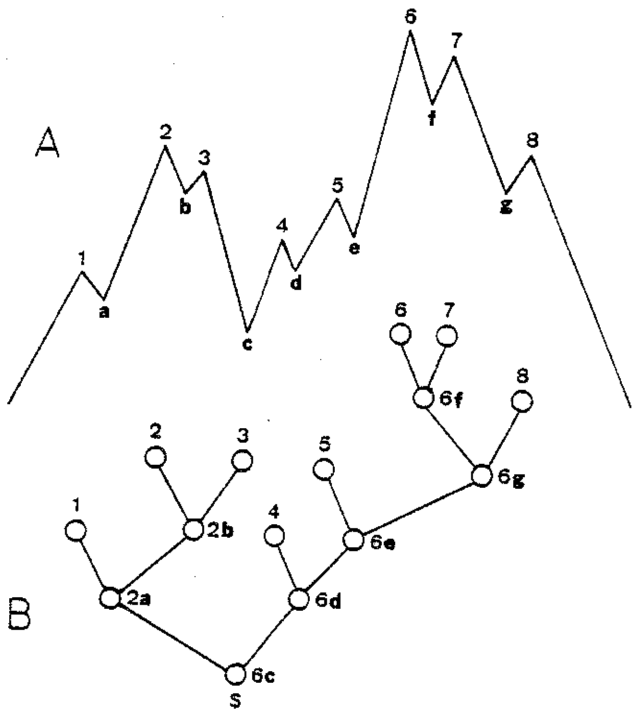


Figure 2 - (a) Sample profile and (b) Its R-tree.

division since it is the lowest minimum value. Each remaining segment is treated as a profile, and lowest valleys (ie. a in the left segment and d in the right segment) are again used for division. Division continues until all valleys of the profile are exhausted, and the recursive structure determined by the partitioning is represented by the relational tree structure. Figure 2a shows a sample profile, and Figure 2b illustrates its relational tree. The root of the tree indicates that over the entire profile the lowest valley is point c and the highest peak is point 6. Since the resultant data structure is a tree, the hierarchical relationships among the peaks given in Figure 2 can be represented in linear form by

(6c(2a(1,2b(2,3)),6d(4,6e(5,6g(6f(6,7),8)))))

or simply by a list structure such as,

((1,(2,3)),(4,(5,((6,7),8)))) .

Thus a relational tree is a complete description of the contextual relationships defined by the peaks and valleys of the intensity profile. Attributes of peaks such as peak height, contrast, etc. can be stored in the tree using pointers. Textural features can then be extracted at any level of the tree. Ehrich and Foith use the R-trees extracted from all scan lines of a textured image as the image representation.

2.2 Two Region Growers

Two different region growers are suggested in this paper. One is based only upon the <u>spatial adjacency</u> relationships among texture elements. These adjacency relationships are represented by the R-tree structure.

Based upon the spatial adjacency relationships among texture elements, the region grower functions by merging microtexture elements into macrotexture elements. It can be described by the following steps:

(1) The original image is transformed into the relational tree data structure.

(2) Terminal nodes in the tree are "pruned" so that the new tree reflects the macrolevel structures of the texture elements.

(3) The image is reconstructed from the pruned tree structure. The reconstructed image will contain larger texture elements that reflect the merged spatially adjacent microtexture elements. that are spatially adjacent to each other.

Steps (2) and (3) are iterated until a level is reached at which global or macro regions appear.

The second region grower is based upon the <u>structural adjacency</u> relationships among texture elements. Structural adjacency refers to the similarity in the attributes of texture elements. Based upon the structural adjacency relationships among texture elements, the region grower looks for similarities among texture elements and groups them together if they are similar. The procedure consists of the following steps:

(1) Construct the relational tree.
(2) Extract texture elements which are stored in the relational tree.
(3) Compute attributes for each texture element.
(4) Define similarity measures.
(5) Group "similar" elements together.
(6) Reconstruct the image by expanding the classified elements to form solid regions over the entire image space.

Zahn [Za71] has found that the <u>minimal spanning tree</u> (MST) for the complete graph of a point cluster is a useful way to extract information about the spatial relationships within the cluster. In the proposed region grower, the MST technique is expanded in order to take care of both structural adjacency and spatial adjacency relationships among texture elements. In effect, we are attempting in this work to make explicit use of the two Gestalt principles of proximity and similarity in growing regions. Zahn's work, in fact, was done with the same intent.

It is well known that Gestalt psychology started with the idea that <u>forms</u> or <u>patterns</u> transcend the stimuli used to create them. The gestalt laws of organization describe groupings or patterns which are most naturally seen as units. Such laws can be used as guidelines while doing texture analysis. However, it is nontrivial to simulate these psychological principles by a computer program, and little work has been done in the application of these principles to texture studies.

3. A Structural Model for Texture

The work presented in this paper is based upon a structural theory of texture. The concept of the relational tree is generalized to two dimensions to account for the structural relationships among texture elements. In this section, the four main aspects of the structural model are discussed in detail. Then the generalized relational tree is presented.

3.1 Details of the Structural Model

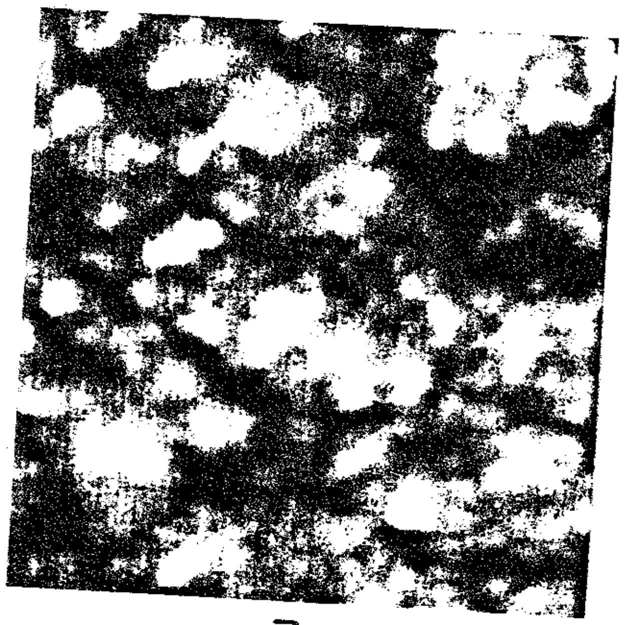
Pure structural models of texture are usually based upon the view that textures are composed of primitives or basic texture elements which are organized according to certain arrangements. Primitives, attributes of texture elements, relationships among elements, and the aggregation mechanisms are four major aspects of the proposed structural texture model.

Theoretically, texture region growing is achieved based upon two assumptions; (1) texture is an area phenomenon that is describable as recursively nested and juxtaposed texture elements, (2) spatial relationships and structural adjacency relationships in attribute space are two basic relationships among texure elements.

3.1.1 Textural Primitives

A primitive is defined as a connected set of picture points characterized by a list of attributes. The simplest primitive is the individual picture point described by its intensity and its location.

It is often desirable to work with larger nontrivial primitives which are maximal connected sets of picture points having some specific property. From these primitives one can extract useful attributes and determine their relationships. Examples of such primitives are connected components of constant gray level, ascending or descending components, saddle components, relative maxima components, relative minima components, etc. Individual picture points are too trivial for this purpose, and there are too many of them to contend with. Instead, we have found that relative extrema are good candidates. If one considers a texture sample such as the one in Figure 3a, the smallest nontrivial perceptual units are the white blobs (or black blobs) from


a

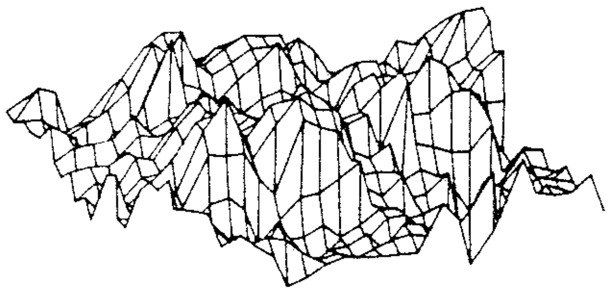

b

Figure 3 - (a) Texture sample and (b) Surface of the picture function for the upper right corner of (a).

which the texture is constructed. In natural textures these blobs are rarely uniform in intensity as shown in Figure 3b, which is a plot of the picture function in the upper right corner of the texture. Therefore the white primitives are taken to be regions consisting of a local intensity maximum together with a local neighborhood that is bounded on all sides by local minima. Such a primitive is called a _structural peak_. The generalized relational tree to be described later is an efficient way for representing such structural peaks.

3.1.2 Attributes of Texture Elements

Each instance of a textural element must be described by a list of attributes that is sufficiently complete to account for perceptual differences among different types of elements. Here one is faced with a potentially infinite pool of features, but there are probably few useful ones. Fortunately, many natural textures have primitives that can be adequately described by a small set of psychologically determined attributes. Examples are gray level local properties such as maximum intensity, minimum intensity, average intensity, contrast, etc. Others include measures of the size, shape, orientation, eccentricity, irregularity, and homogeniety of the texture elements.

When extracting extrema in one dimension, simple properties such as _height_ and _width_ can be associated with each extremum. The absolute height of a peak is defined as the maximum value, and the relative height can be defined as the intensity difference between the maximum and the average of its adjacent minima. The width of a peak is the horizontal distance between its two adjacent minima.

If one considers extrema in two dimensions, then the situation becomes much more complicated. One way of extracting the 2D extrema is by using the generalized relational tree algorithm which will be described later. More complex attributes can be measured for 2D extrema. The relative height of a structural peak is the intensity difference between its relative maximum and the average of its exterior border points. Its size can be defined as the number of picture points in the mountain, or the volume of the mountain.

3.1.3 Relationships Among Elements

There are two classes of relationships among texture elements. The first class consists of hierarchical "containment" relationships among texture elements. These are called spatial adjacency relationships, and they are represented by the relational tree structure. The second class of relationships consist of those that are induced by external constraints. These are called structural adjacency relationships, and they include relationships due to similarities among texture elements. In this work structural adjacency relationships are assumed to exist only at the frontier of the spatial adjacency hierarchy. For example, in Figure 4 the dotted lines constitute the structural adjacency relationships among the frontier elements in the R-tree, while the spatial adjacency relationships are reflected in the R-tree itself.

Spatial adjacency relationships are hierarchical; they are intuitive and not difficult to understand, and they are naturally embedded in the R-tree structure. Figure 5 shows some examples of such relationships. Pictures 5a-5c were produced from the same negative by varying printing

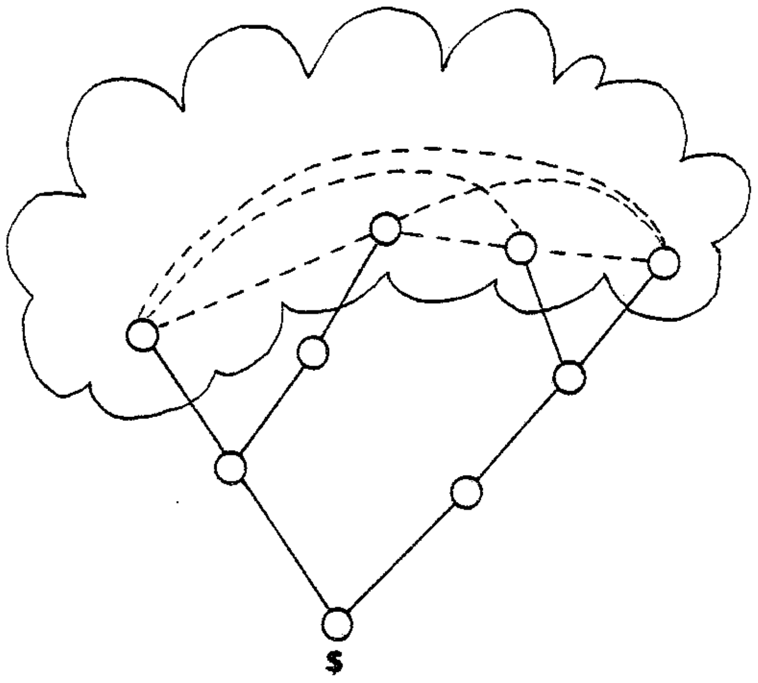


Figure 4 - Spatial adjacency and Structural adjacency relationships among texture elements.

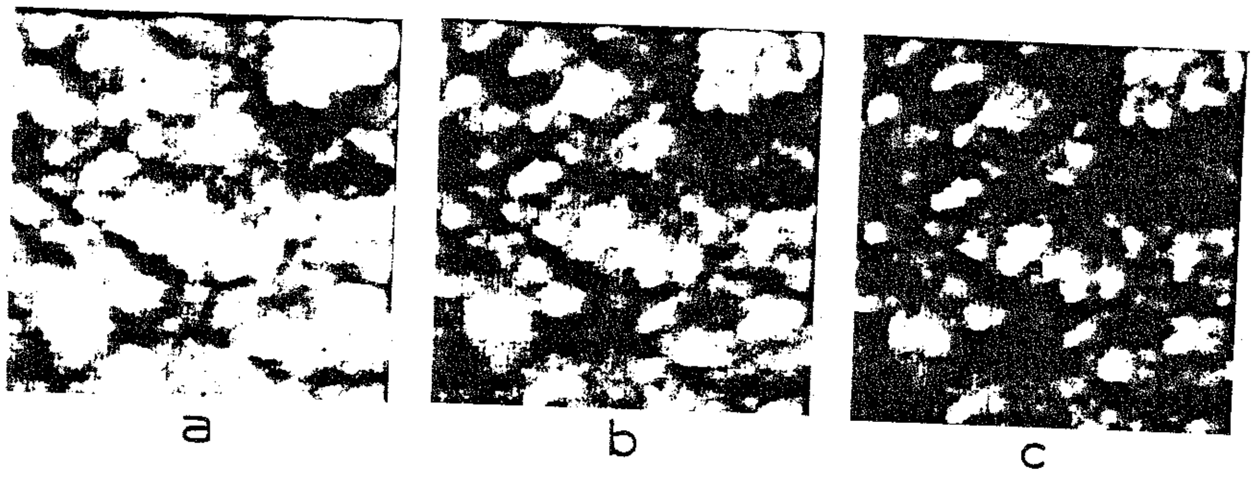


Figure 5 - (a)-(c) Three pictures of RUG produced from the same negative with decreasing printing time.

time to show the details of texture elements in different gray level ranges. Notice that some of the white blobs that appear distinct in picture 5a appear merged in picture 5c. At the same time, new white blobs apear in picture 5c which were not visible in picture 5a. The small white blobs in picture 5a are examples of <u>microtexture</u> <u>elements</u>, and the larger blobs in picture 5c are examples of <u>macrotexture</u> <u>elements</u>. It is important to keep in mind that both macrotexture and microtexture are simultaneously visible, and there is no basis for ignoring either.

In the strict sense, every texture element is structurally related to every other one. Conceptually, these relationships can be represented by a <u>complete</u> <u>weighted</u> <u>graph</u> called the <u>texture</u> <u>graph</u>. Vertices of the graph are instances of texture elements, and each edge is weighted by a distance or a weighting function between the pair of elements it links. The weight between a pair of elements, in turn, depends upon attributes of the elements. This graph will be an essential part of a region grower to be described later.

<u>3.1.4</u> <u>Aggregation</u> <u>Mechanisms</u>

Aggregation is one of the least explored aspects of computer vision but has great importance in determining interpretations of visual stimuli. Aggregation in this paper is based entirely upon the assumption that no prior knowledge is available about the textures present in a scene. What is significant about the structural model of texture used here is that the psychological factors appear only in this fourth aspect of the structural model and do not become confused with measurement processes. Two kinds of aggregation processes can be formulated based upon the spatial and the structural adjacency relationships. The first of these is based upon the containment relationships that explicitly exist in the relational tree. Pruning frontier nodes from the tree structure has the effect of merging small peaks into larger ones in much the same way as that demonstrated in Figure 5. The second aggregation mechanism is based upon structural adjacency relationships among texture elements. The texture graph suggests that every texture element is structurally related to every other texture element. The closeness of these relationships will be determined by the similarities of the attributes of each pair of texture elements, and the graph edges will be weighted by measures of attribute similarity.

3.2 Generalized Relational Tree

Ehrich and Foith [Eh76] were the first to propose a hierarchical structural representation of an intensity profile in terms of its peaks and valleys. Based upon the hierarchical relationships among peaks, a tree called a relational tree is generated for each profile. Recently, Rosenfeld [Ro77] developed a theory called "fuzzy digital topology," in order to account for the topological relationships among portions of a gray level image. In particular cases, Rosenfeld's theory also will account for the relationships among peaks and valleys of the two-dimensional picture function. The application of the theory to one-dimensional profiles is developed by Sankar and Rosenfeld [Sa79]. The approach proposed by Rosenfeld and Sankar is elegant in terms of using concepts of fuzzy connectivity; however, it is hard to visualize intuitively.

In this section, the concept of the relational tree is generalized to two-dimensions. In principle, the generalized two-dimensional R-tree is similar to the "image tree" suggested by Krakauer [Kr71] except that the algorithm used to generate tree nodes and the features extracted from the tree are different. The principles that the new algorithms are based upon are different from those in Rosenfeld in the sense that the new algorithms make use of simple topological properties of a binary image.

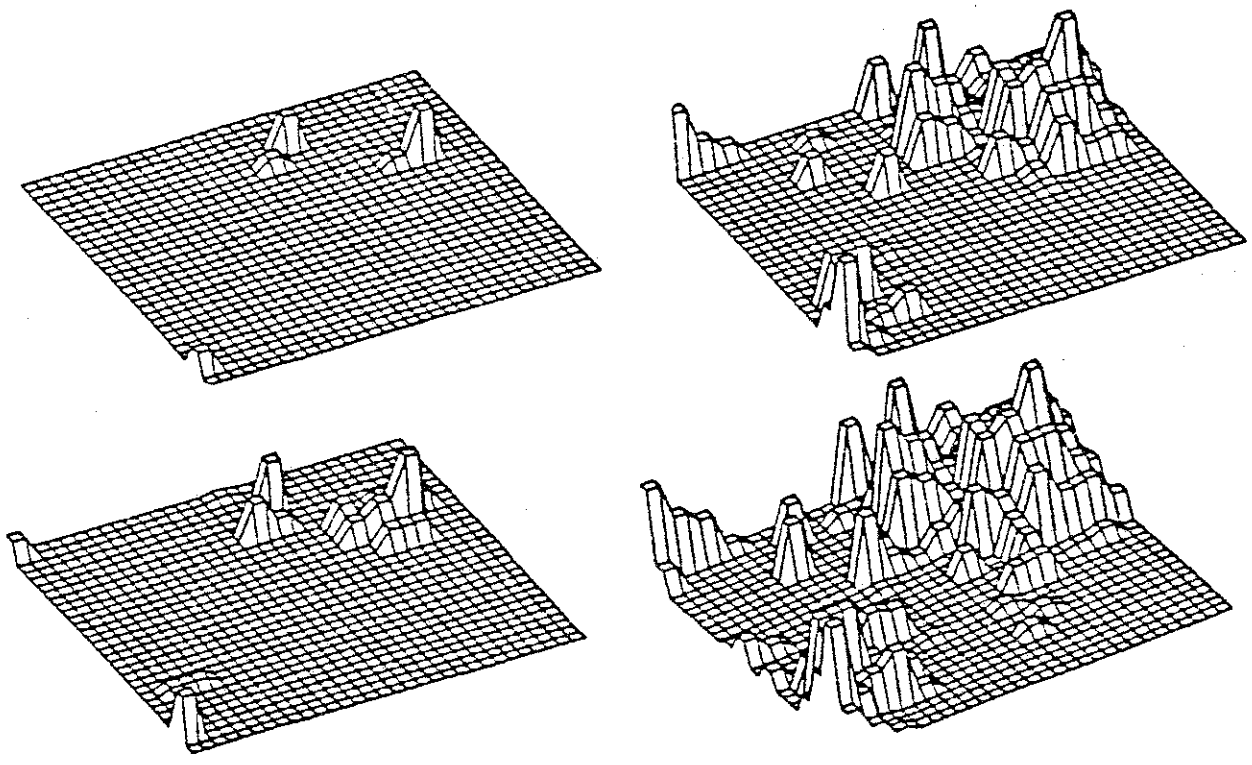


Figure 6 - A visualization of the two-dimensional R-tree.

As in Figure 6, imagine a fixed plane through which the topological surface of the picture function is pushed from below. As the highest peaks of the picture function penetrate the plane, the isolated volumes above the plane define peaks that correspond to the strongest white stimuli. At this time, one could generate an external node for each such peak. As the picture surface is pushed farther into the plane, peaks begin to merge or to expand or new peaks start to emerge. Whenever two or more peaks merge, one generates an internal node to represent the hierarchical relationship between the new peaks and their subpeaks. Continuing in this way, as the lowest valleys of the picture function pass through the plane, one eventually generates a root node for this super peak. Therefore, the two-dimensional peaks and valleys fall naturally into a tree structure, which is called a _generalized relational tree_.

Let S be a rectangular array of discrete coordinates, and f: S -> {0,1} be a binary picture function. Obviously any point P at coordinate (x,y) has 4 horizontal and vertical neighbors, namely (x+1,y), (x-1,y), (x,y+1), and (x,y-1). P also has 4 diagonal neighbors, namely (x-1,y-1), (x-1,y+1), (x+1,y-1), and (x+1,y+1). A _path_ between two points p and q is defined as a sequence of points $p=p_0, p_1, \ldots, p_n=q$ such that p_i is _adjacent_ to p_{i-1}, where $1 \leq i \leq n$. p and q are _connected_ if there is a path from p to q and $f(p), f(p_1), \ldots, f(q)$ all have value 1. Notice that path and connectedness can be defined based upon either 4-adjacency or 8-adjacency depending upon whether or not one considers the diagonal neighbors. Connectedness is an equivalence relation that partitions S'

$= \{(x,y) \in S \mid f(x,y)=1\}$ into equivalence classes $S'_1, S'_2, \ldots, S'_n$. These classes are called the <u>connected</u> <u>components</u> of S'.

Now consider a picture function that has L gray levels. Let f: S -> [0,L-1] be the picture function. Define S_i as the set of picture points whose intensity is greater than or equal to gray level L-i; that is,

$$S_i = \{ (x,y) \mid f(x,y) \geq L-i \}$$

In particular,

$$S_0 = \{ (x,y) \mid f(x,y) \geq L \}, \text{ and}$$

$$S_L = \{ (x,y) \mid f(x,y) \geq 0 \}.$$

Obviously, S_0 is an empty set, and S_L is the set containing all picture points, which is the set S.

Let f_i be a binary function which maps all points from S_i to 1, and other points to 0. In other words, $S_i = \{(x,y) \in S \mid f_i(x,y)=1\}$. Each S_i can be partitioned into disjoint subsets $S_{i,1}, S_{i,2}, \ldots, S_{i,R(i)}$ based upon the connectedness relation. The subsets S_i have the following properties:

PROPERTY 1:

$$S_0 \subseteq S_1 \subseteq \ldots \subseteq S_L = S.$$

PROPERTY 2:

Let $S_i = S_{i,1} \cup S_{i,2} \cup \ldots \cup S_{i,R(i)}$ have R(i) components, and let $S_{i+1} = S_{i+1,1} \cup \ldots \cup S_{i+1,R(i+1)}$ have R(i+1) components. Then for any component $S_{i+1,j}$ of S_{i+1}, one of the following three statements is true.

(1) $S_{i+1,j} \cap S_i = \emptyset$,

(i.e. $S_{i+1,j}$ is a new component).

(2) $S_{i+1,j} \cap S_i = S_{i,k}$, where $1 \leq k \leq R(i)$,

(i.e. component $S_{i+1,r}$ is the expansion of $S_{i,k}$).

(3) $S_{i+1,j} \cap S_i = S_{i,k1} \cup S_{i,k2} \cup \ldots$,

where $1 \leq k1,k2,\ldots, \leq R(i)$,

(i.e. component $S_{i+1,j}$ contains subcomponents $S_{i,k1}, S_{i,k2}, \ldots$).

If one defines a plateau as a maximal connected subset of S such that its f value is constant, a peak as a plateau whose f value is a local maximum, and a valley as a plateau whose f value is a local minimum, one has:

PROPERTY 3:

In case 1 of Property 2, $S_{i+1,j}$ is a peak in S.

PROPERTY 4:

In case 2 of Property 2, $S_{i+1,j}$ is a peak slice.

PROPERTY 5:

In case 3 of Property 2,

let $S_{i,K} = S_{i,k1} \cup S_{i,k2} \cup \ldots$

then $S_{i+1,j} \cap \overline{S}_{i,K}$ contains some valleys in S.

Based upon Properties 2-5, an algorithm for constructing a relational tree can be devised. It is given by the following steps:

```
ALGORITHM 1:
(1) Set $i=0$.
    Extract $S_i$ from S.
    Extract all disjoint components from $S_i$.
    Generate an external node for each of the components.
(2) Repeat steps 3-5 while $S_{i+1} \subset S$.
(3) Extract $S_{i+1}$ from S.
    Extract all disjoint components from $S_{i+1}$.
    Set $j=1$.
(4) Repeat steps 4.0 through 4.4 while $j \leq R(i+1)$.
    (4.0) Set $T=S_{i+1,j} \cap S_i$.
    (4.1) If $T=\emptyset$, then generate an external node for $S_{i+1,j}$.
    (4.2) If $T=S_{i,k}$ then stack component $S_{i+1,j}$ with $S_{i,k}$.
    (4.3) If $T=S_{i,k_1} \cup S_{i,k_2} \cup \ldots$, then generate an intenal node for
    $S_{i+1,j}$ and link all possible nodes generated for $S_i$ to this node if
    their corresponding components are contained by the component
    $S_{i+1,j}$.
    (4.4) Set $j=j+1$.
(5) Set $i=i+1$.
(6) Generate the root node (for S), link all subcomponents to it, and
    terminate the procedure.
```

At the same time that a tree node is generated, information about the peak's relational structure and its attributes can be computed. For example, an external node defines a peak as it is first generated in Step 4.1. In Step 4.2, information about this particular peak is accumulated. In Step 4.3, a peak merges with other peaks, thus inducing an internal node, peak volume can be computed by accumulating the areas of all the descendant nodes. Other attributes such as center of mass and relative height can also be computed. Therefore, a well-defined peak in the three-dimensional X-Y-INTENSITY space can be extracted. This process can be carried out down to any internal node to form a more complicated peak structure.

4. Texture Element Extraction

The two-dimensional R-tree algorithm given in Section 3 consumes considerable computational resources. In the case of the region grower based upon structural adjacency relationships, only the primitives at the frontier of the R-tree are considered. Therefore it is not necessary to compute all of the deep structure of the R-tree, and two approximate methods can be used. In the first method, frontier peaks are computed approximately by using a local peak extractor, and in the second method, an asymptotic relational tree is computed for which horizontal crossections of peaks are computed only for a small number of different intensities.

4.1 Primitive Extraction by Local Peak Detection

A simple 3x1 local operator can be easily designed to detect local maxima and local minima of a one-dimensional scan line simultaneously in one pass. One has to be careful to handle flat (plateau) regions properly, and in this first experiment, peak-valley labels were assigned as shown in Figure 7 to facilitate a one-pass, left to right analysis. This one-dimensional operator is then applied to image intensity profiles row by row and column by column to locate picture points that are extrema simultaneously in both directions.

In one dimension, peak intensity is measured by its absolute height. Peak contrast is measured by its relative height, and peak size can be approximated by its width. Let AH_x, RH_x, and W_x denote the corresponding one-dimensional attributes in the horizontal direction, and let AH_y, RH_y, and W_y be the attributes in the vertical direction.

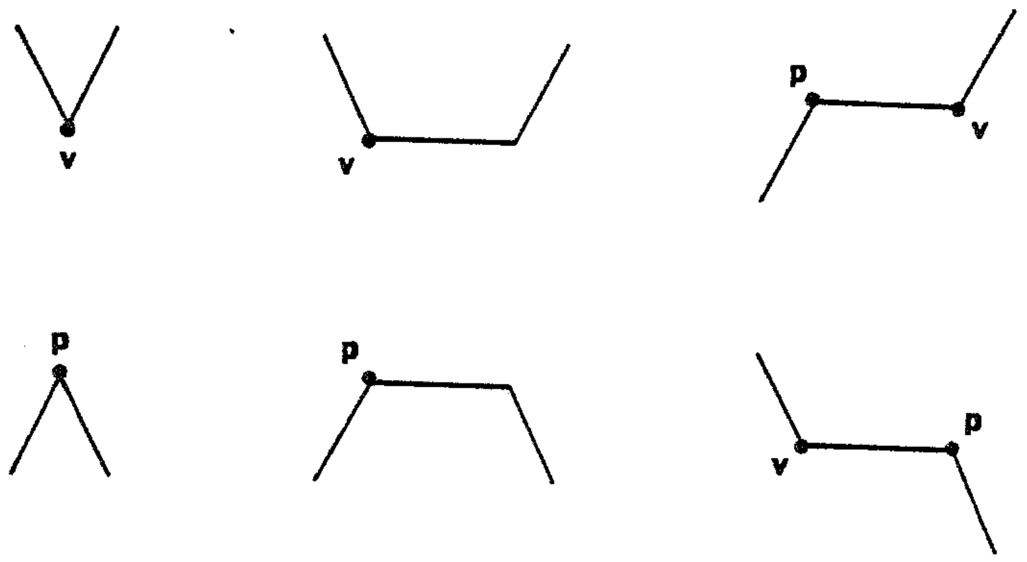


Figure 7 - Peak-valley label assignments.

Let X_p be the horizontal location of a peak, and let X_u, X_v be the locations of its left and right valleys, respectively. Then, the absolute height and the peak width are defined by

$$AH_x = f(X_p), \quad W_x = | X_u - X_v |$$

and the relative height is defined by

$$RH_x = f(X_p)-((f(X_u)+f(X_v))/2 .$$

Figure 8 illustrates these definitions. The two-dimensional relative height can be computed as the average of the relative heights in the two

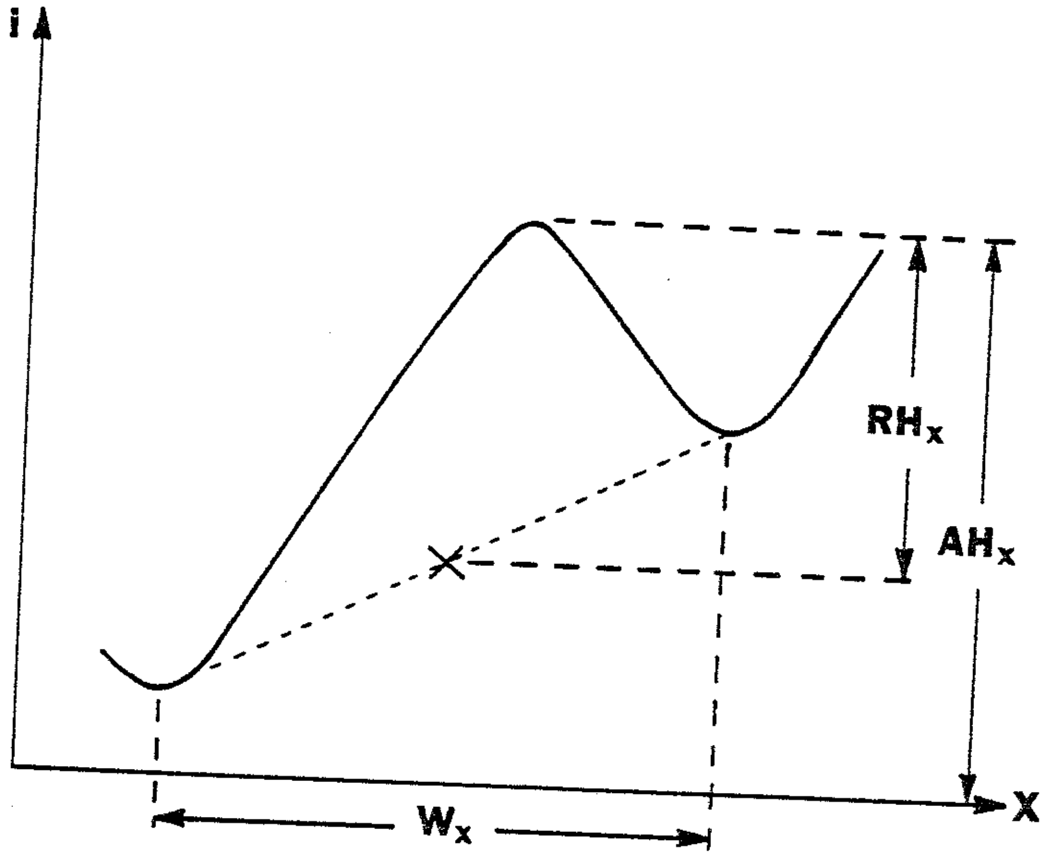


Figure 8 - One-dimensional peak attributes.

directions. Peak area can be approximated by taking the product of the peak widths in the two directions. Let AH, RH, and SZ denote the absolute height, relative height, and the peak area in two dimensions, respectively. Then,

$$AH = AH_x = AH_y$$
$$RH = (RH_x + RH_y)/2$$
$$SZ = W_x W_y$$

Also, shape measures such as elongation can be computed by

$$EL = \tan^{-1}(W_y/W_x) \ .$$

4.2 Asymptotic Relational Trees

Asymptotic R-trees (or simply, ART's) are approximations to true two-dimensional R-trees in which texture elements are described by a stack of approximate cross-sectional slices of the picture function. Each such slice forms a binary image in which a picture element is set to 1 if the corresponding image intensity is larger than the slice intensity and 0 otherwise. For the purpose of representing a texture, an image is thresholded at a number of selected values, and an algorithm described by Wang [Wa78] is applied for extracting all the connected components (regions or holes) of the resulting binary images. Then the relations among the connected components of the various slices are determined, and the R-tree is generated. At the same time, attributes are determined for the various texture elements and stored in the data structure.

Wang's procedure performs boundary smoothing and hole filling on each connected component. It also produces for each binary image an attribute table that specifies for each component its area and its regularity. Each component is also related to the original image by computing the intensity extremes, contrast, and average intensity over the area of the component. Figure 9 shows an example of the components produced from Figure 3 by Wang's algorithm with a particular choice of threshold.

Let S be the discrete coordinate set of an image array, and let f: S-> { 0,1,...,L-1 } be the picture function with L gray levels. Define the set S_i by

$$S_i = \{ (x,y) \mid f(x,y) \geq L-i \}.$$

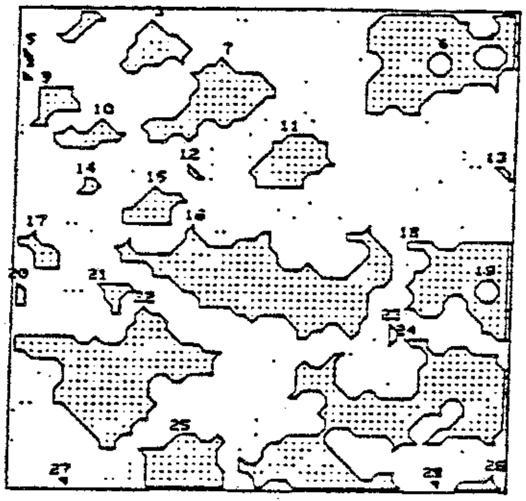


Figure 9 - Components extracted from Figure 3 by Wang's algorithm at a particular threshold.

According to the algorithms described earlier, in order to compute the exact relational tree, one has to extract and find the relationships among the regions of two successive S_i.

To find an asymptotic relational tree, instead of working on all gray levels, one needs to select a subset of gray levels at which to threshold the original image. Let $R = \{0,1,\ldots,L-1\}$ be the set of available gray levels, and let $R' = \{I_1,I_2,\ldots,I_K\} \subseteq R$ be the selected subset. A simple scheme to determine this subset is to find R' such that it divides the gray level histogram into equal area portions. Let T_i be the set of picture points whose gray level is greater than I_i, and let Q_i be the set of picture points whose gray level is between two successive thresholds, namely I_{i-1} and I_i. That is,

$$T_i = \{ (x,y) \mid f(x,y) \geq I_i \}$$

$$Q_i = \{ (x,y) \mid I_{i-1} \geq f(x,y) \geq I_i \} .$$

Each Q_i corresponds to a portion whose size is 1/K of the size of the entire image array, and each T_i corresponds to the accumulation of $Q_1, Q_2, \ldots, Q_i$ whose size is therefore i/K of the size of the image array. These facts are summarized by the following properties:

PROPERTY 6:

(i) $|Q_1|=|Q_2|=\ldots=|Q_K|=|S|/K$.

(ii) $T_i=T_{i-1} \cup Q_i$, where $1 \leq i \leq K$.

(iii) $|T_i|=(i/K)|S|$.

The subset R' attempts to select the intensity thresholds so that T_1 contains the 1/K brightest elements of the image array, T_2 contains the 2/K brightest elements, and so on. This subset of thresholds is commonly called the _equal-probability quantizations_. Each T_i can be partitioned into disjoint components called _the intensity sliced regions_ at intensity I_i, or at slice i, and the corresponding thresholded image is also called the _intensity sliced image_. Algorithm 1 can be modified to generate a tree structure based upon K levels of quantization. This tree structure is called the _asymptotic relational tree_ (ART). Besides the reduced number of quantization levels, another difference between the ART and the true R-tree is that in the ART, external nodes do not always represent a peak structure because some peaks merge between two successive slices. The following is the modified procedure for constructing the ART:

ALGORITHM 2:

(1) For a given intger K, compute $R'=\{I_1,I_2,...,I_k\}$.

Set i=1.

Extract T_i by thresholding the picture function f at I_i.

Extract all components detected in T_i.

Generate an external node for each of the components.

(2) Repeat steps 3-5 while $i \leq K-2$.

(3) Extract T_{i+1} (by thresholding).

Extract all components detected in T_{i+1}.

Set j=1.

(4) Repeat steps 4.1 through 4.3 while $j \leq R(i+1)$.

(4.1) If $T_{i+1,j}$ is a new component, then generate an external node for this component.

(4.2) If $T_{i+1,j}$ is not a new component, then generate an an internal node for $T_{i+1,j}$ and link all possible nodes generated for T_i to this internal node if their corresponding components are contained by the component $T_{i+1,j}$.

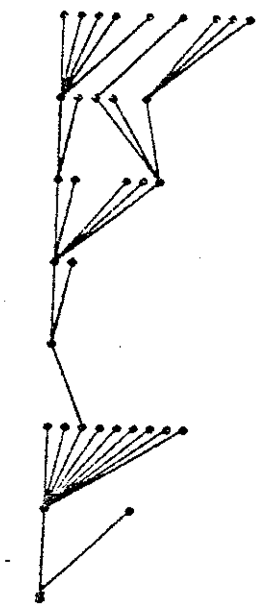
(4.3) Set j=j+1.

(5) Set i=i+1.

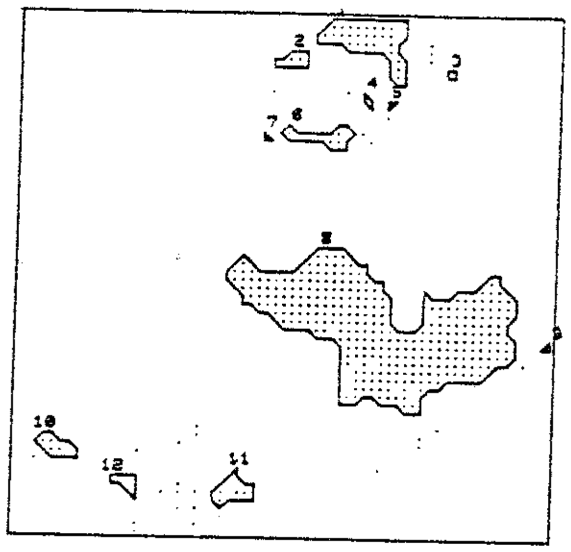
(6) Generate the root node for T_k, link all nodes generated for T_{k-1} to the root node, and terminate the procedure.

An example is shown in Figure 10 with K=8 and

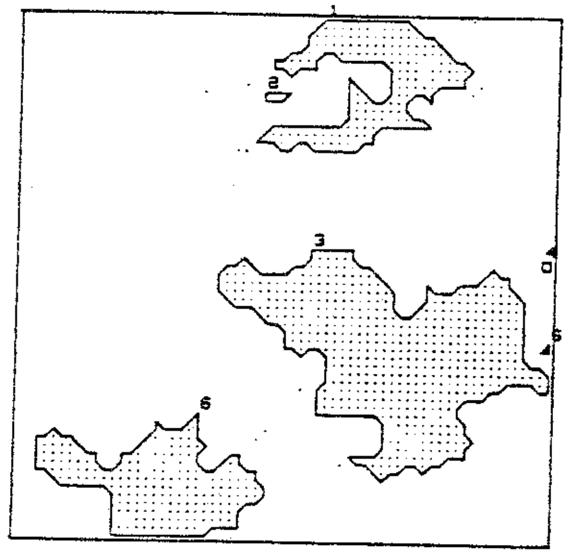
R'={223,220,216,211,202,171,166,154}.



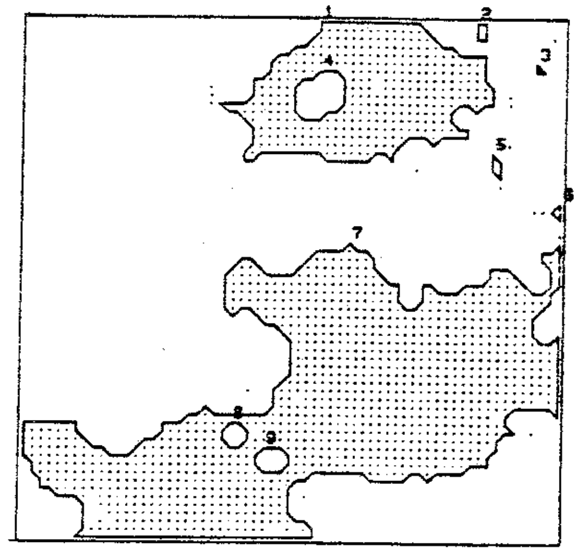
a



b

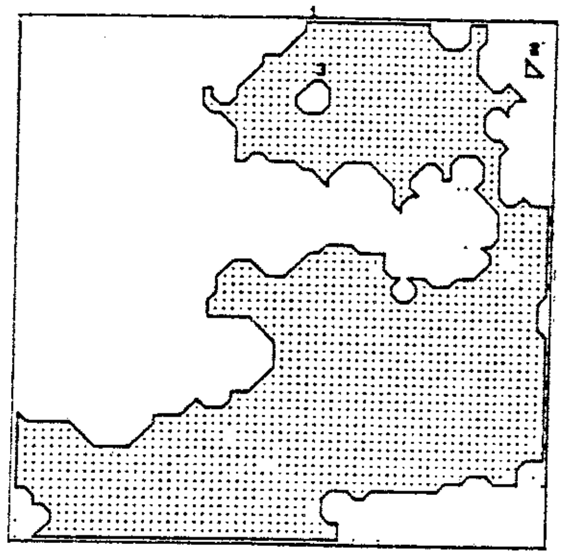


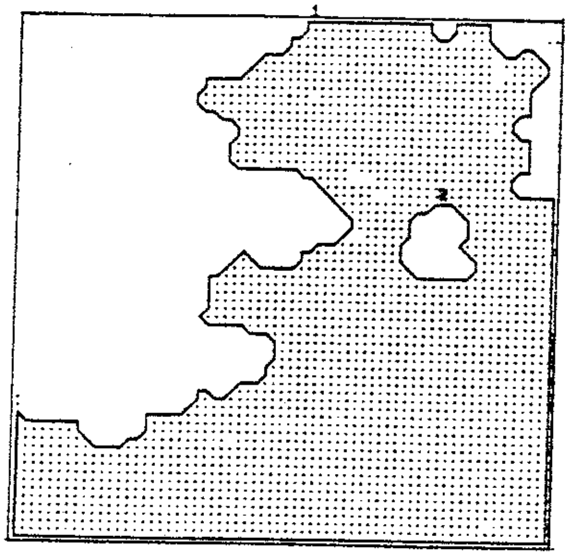
c

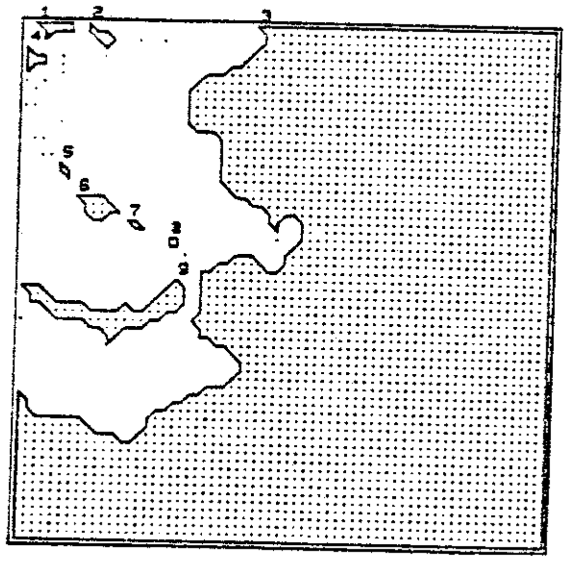


d

Figure 10 - (a) ART for SKY-CLOUD sample with 8 thresholds and (b)-(d) Image thresholded at 223,220,216.


e


f


g

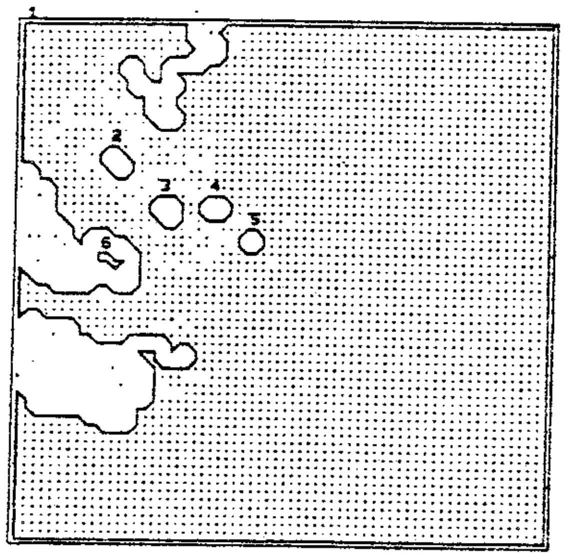

h

Figure 10 (cont) - (e)-(h) Image thresholded at 211,202,171,166.

5. Texture Region Growing Using Minimal Spanning Trees

This section is devoted to the development of a texture region grower that is based upon structural adjacency relationships among texture elements. The basic assumption is that "similar" elements are the ones that ought to be grouped. The method to be used here employs minimal spanning trees (MST) in much the same way as Zahn [Za71], except that the edge weights in the texture graph are computed by similarity measures on a multidimensional attribute space.

5.1 Attribute selection

One would like to have available as many descriptive attributes for texture elements as possible. These attributes can be stored in the data structure to produce a "complete" description of each element. However, it is not well understood which of them are important in texture analysis, since there are so many measurable attributes and so many possible relationships among the texture elements. The problem hinges upon psychological mechanisms. Zobrist and Thompson [Zo75] performed psychological tests to determine the parameters of a distance function that was used to simulate human perceptual grouping. Their experiment suggests a way of combining multiple cues. A primitive distance function is used for each cue to measure the strength of similarity grouping. The total tendency of two regions to be grouped together is measured by a linear weighted sum of the primitive distance functions, and the weights are determined by psychological tests. Therefore, a distance function for Gestalt grouping on textures can be "built up" through psychological testing. However, the particular function developed under certain conditions may not be useful under others. The approach in this work is to try to determine first which attributes of texture elements are the best among those available. This is followed by the determination of a distance function on the selected attributes. One possible way of selecting the attributes is called automatic attribute filtering.

The goal of automatic attribute filtering is to eliminate from further consideration those attributes that do not have a significant role in the perception of a particular textured image. For a given texture sample, it is possible to compute some simple statistics for each attribute to determine the dominant ones. Then grouping is done based upon the selected attributes. Since the number of texture elements is much smaller than the number of picture elements in the original image array, it is not impractical to compute such statistics. This procedure resembles the use of redundancy in human perception in the sense that the human visual system appears to have the ability to adaptively select appropriate cues from a large pool of cues.

The statistical measures used for attributes were the sample mean, sample deviation, and ranges. Assume there ar N texture elements. Let $A_j(i)$ be the jth attribute of the ith texture element. For the jth attribute, the sample mean is

$$M_j = (A_j(1)+A_j(2)+\ldots+A_j(N))/N$$

and the sample deviation is

$$V_j = |A_j(1)-M_j|+\ldots+|A_j(N)-M_j| \ .$$

The range is defined by

$$R_j = \mathrm{Max}\{A_j(1),\ldots,A_j(N)\} - \mathrm{Min}\{A_j(1),\ldots,A_j(N)\}.$$

Then, a busyness measure, called the <u>degree of importance</u> for the jth attribute is defined by

$$\mathrm{THETA}_j = V_j/R_j \ .$$

THETA is used as the measure for determining the important attributes, and it is computed for each of the available attributes. Then the important attributes are determined by selecting those with larger THETA values. In defining THETA, R_i is a normalization factor. A flat histogram of attribute values tends to have lower deviation and would have a lower THETA value. The more modes that exist in a histogram, the higher the deviation will be, which would result in a higher THETA value. Figure 11 shows THETA values and the associated histogram for four attributes of a texture sample. The four attributes are peak height, peak contrast, horizontal peak width, and vertical peak width. The corresponding THETA values are .322, .09, .108, and .093, respectively. These values indicate that the peak height is the most important attribute. In fact, the texture is the SKY-CLOUD sample which is composed of two natural textures which differ mainly in their intensity.

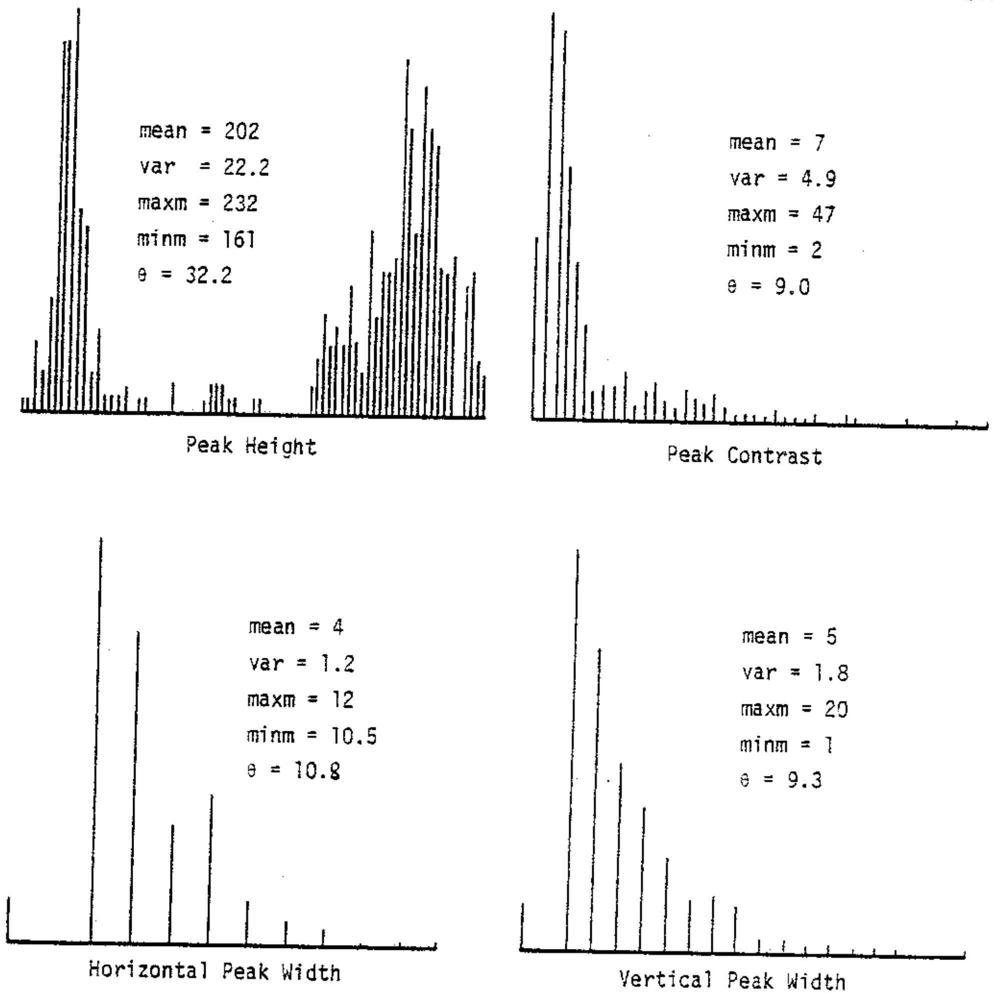


Figure 11 - THETA values and the corresponding histograms.

Having filtered out some of the less important attributes, the next step involves selecting a distance function for each remaining attribute. This is a difficult problem because any such distance function is a parameterization of psychological mechanisms that are poorly understood. One approach might be to assume the functional form of the distance function and to optimize it by selecting the parameters on the basis of psychological evidence as in Zobrist and Thompson.

Assume there are K important attributes. Let $f_k(i,j)$ be the individual distance function for the kth attribute between texture elements i and j. Then the overall distance can be defined by

$$d(i,j) = f_1(i,j) + f_2(i,j) + \ldots + f_K(i,j) .$$

Such an approach does not guarantee a solution to the original texture problem no matter how well the distance function has been optimized because not even the best functional form is known. Another approach might be to solve the problem once for each attribute by itself and to combine the results later. Such an approach would eliminate the problem of determining the best combination of the primitive distance functions at the expense of having to combine the individual grouping results later. The approach used here is to use the same functional form as in Zobrist and Thompson; however, the individual distance functions are fixed for all attributes.

<u>5.2 The Region Grower</u>

Once the distance function is determined, a minimal spanning tree can be constructed for the texture graph. For the purpose of texture region growing, the MST algorithm is extended to both spatial coordinates and attribute space. One may also apply the MST technique to the attribute space first to determine an initial partition and then apply the MST technique to the Euclidean space to obtain finer partitions. The algorithm has three main steps.

(1) Construct the MST using Prim's algorithm [Pr57].

(2) Divide the MST into fragments by deleting inconsistent edges.

(3) Reconstruct the segmented regions that correspond to the tree fragments.

The MST of the texture graph tends to connect the closest and therefore the most similar texture elements together. The goal is to break the MST at selected edges so that the texture elements in the same fragment bear a closer resemblance to one another. By definition, if one deletes an edge from a tree, the tree is broken into two subtrees, and the corresponding vertices are partitioned into two disjoint subsets. Then, the segmented regions of texture are reconstructed from the partitioned subtrees. The hard problem is that of determining which edges of the MST should be deleted.

An edge is called an <u>inconsistent edge</u> if its length is not consistent with those in its neighborhood; inconsistent edges are those to be considered for deletion. Long edges and inconsistent edges are not necessarily the same. Consider, for example, the longest edge e shown in Figure 12; deletion of e does not give us a "reasonable" partition. This suggests that an "absolutely long" edge may not be a good candidate for an inconsistent edge. "Relatively long" edges are better choices for inconsistent edges. Deleting edge e' in Figure 12 gives us a more plausible partition. The degree of edge inconsistency is defined as the ratio of the edge weight to the average weight of nearby edges. Edges with a high degree of inconsistency are candidates for deletion. At the same time, class labels are assigned to the texture elements associated with each tree fragment.

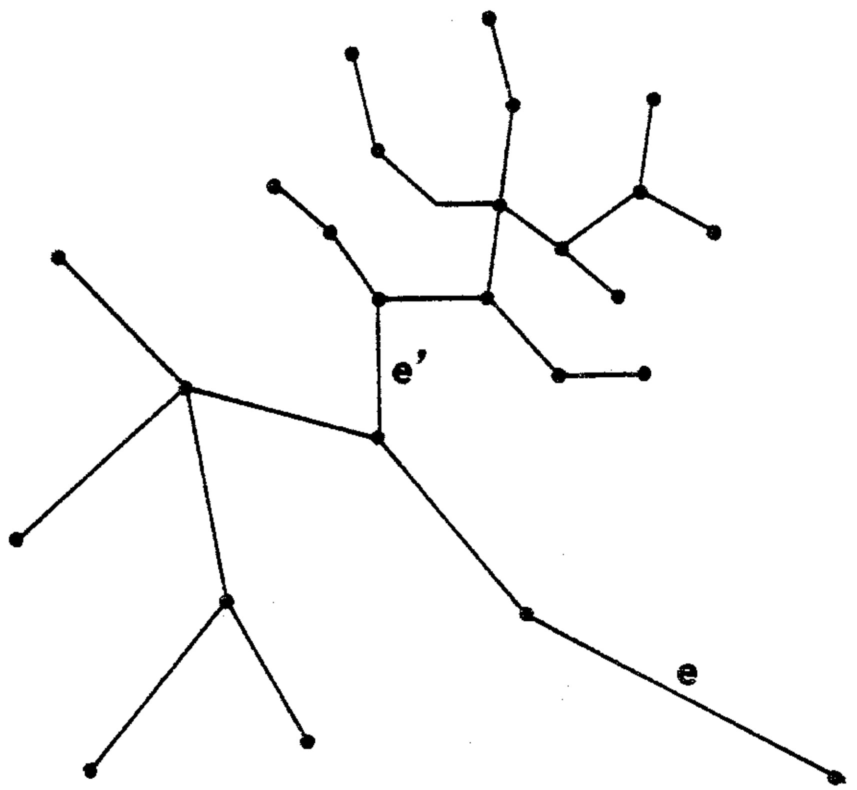


Figure 12 - Illustration of a "long" edge (e) and a "relatively long edge" (e').

The final problem is to label each picture element of the image array with the class labels in such a way that the boundaries between different textures are consistently formed. This problem is equivalent to the pattern recognition problem of determining decision surfaces that separate the clusters of points with different labels. A straightforward method is simply to assign a point the labels of its nearest texture element. A fast algorithm, called a "diffusion algorithm," proposed by Lai and Ehrich [La79], is used in this work.

5.3 Results Using Local Extrema

Experiments were run on 8 texture samples of 64 x 64 picture elements each using the local extrema extraction technique. The available attributes and their corresponding primitive distance functions are

(1) peak intensity: $f_p(i,j) = (AH(i)-AH(j))^2$

(2) peak contrast: $f_c(i,j) = (RH(i)-RH(j))^2$

(3) peak size: $f_s(i,j) = (SZ(i)-SZ(j))^2$

(4) peak locations: $f_x(i,j) = |x(i)-x(j)|$

$f_y(i,j) = |y(i)-y(j)|$.

The edge weights in the texture graph were computed by using distance functions that were sums of the various primitive functions given above.

In the experiments, the "nearby edges" used for the edge inconsistency test are defined as the edges which can be reached within two steps of a given edge. An edge is broken whenever the degree of inconsistency is greater than a threshold T. Next, region labels are assigned to the texture elements of each tree fragment, and the texture region is filled using the diffusion algorithm.

In the first example, the SKY-CLOUD texture is shown. In the first test, peak intensity, contrast, and spatial location were selected, and in the second test, location information was deleted. Figure 13a shows the original texture, and Figure 13b shows the MST constructed using all three attributes. Notice that the edge lengths in the MST do not correspond to the value of the distance function because the graph has been projected onto two-dimensional space for display. Figure 13c shows

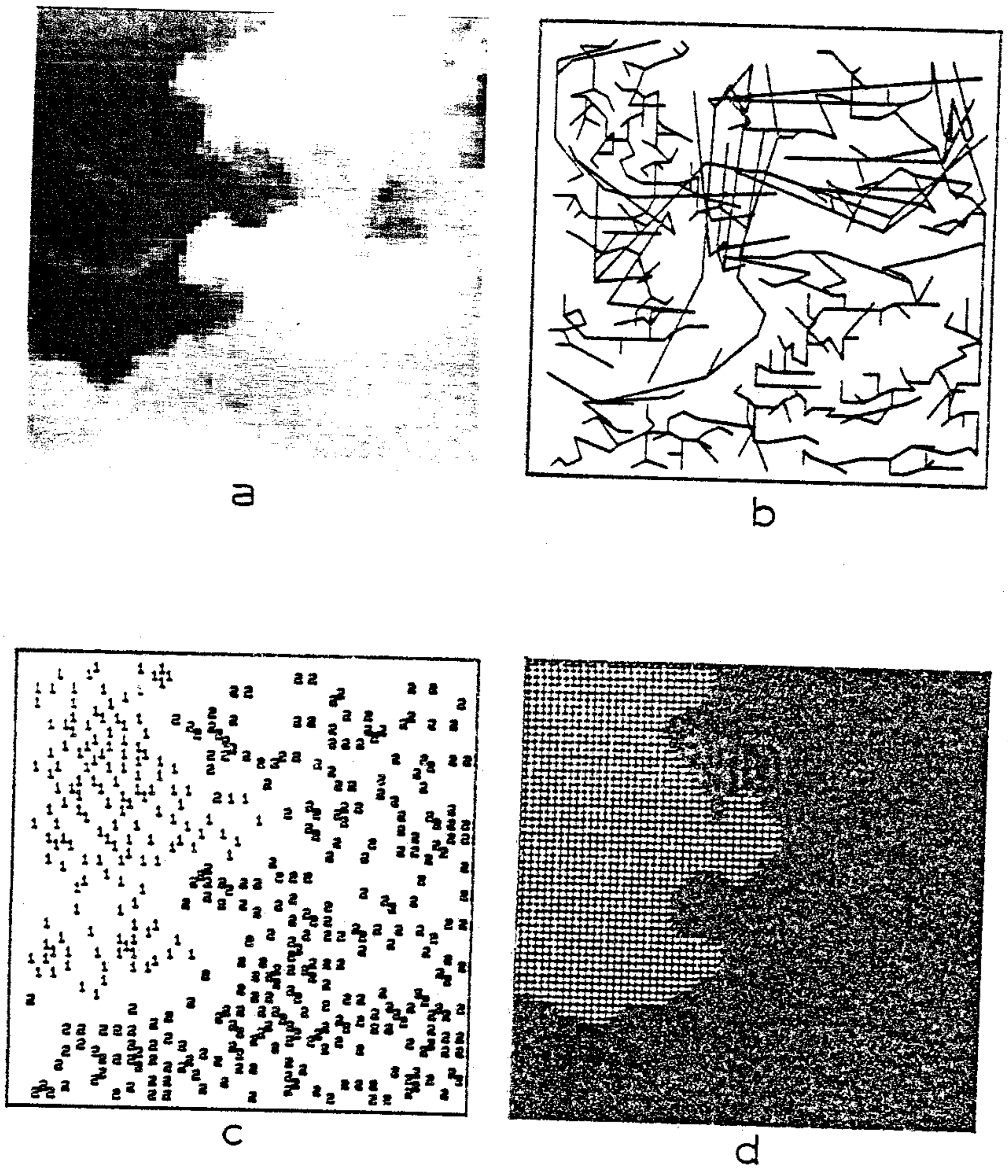
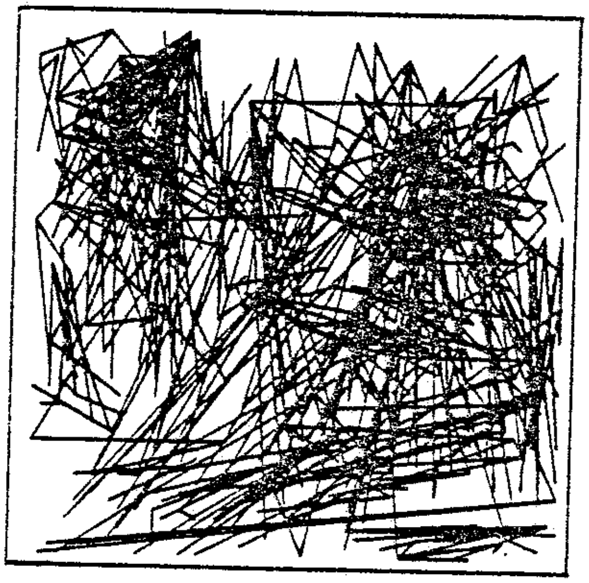
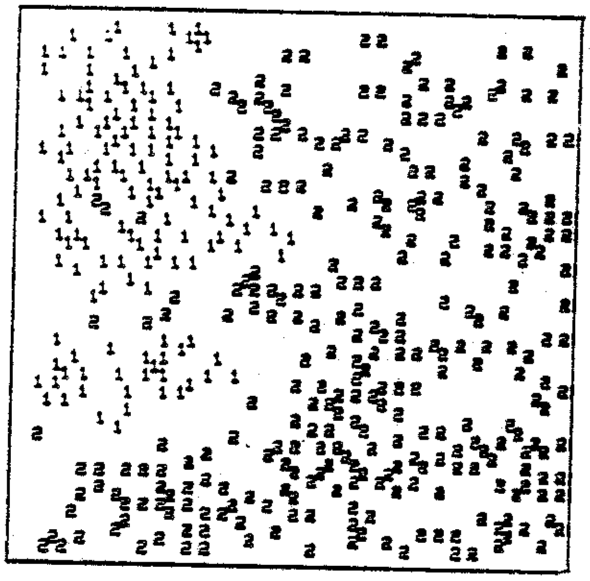


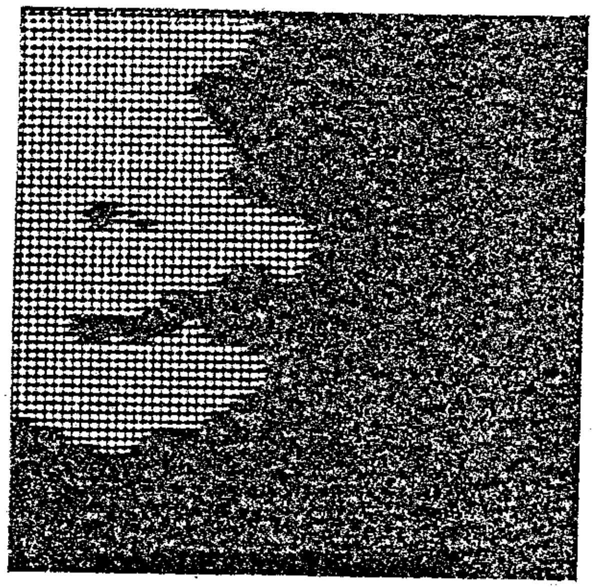
Figure 13 - (a) Original SKY-CLOUD sample, (b) MST, (c) Labeled primitives, and (d) Reconstructed regions.



e



f



g

Figure 13 (cont) - (e) MST without proximity measure, (f) Labeled primitives, and (g) Reconstructed regions.

the labeled elements after deletion of the most inconsistent edge, and Figure 13d shows the result after application of the diffusion algorithm. Figures 13e - 13g show the result obtained using the same procedures as in Figures 13b - 13d except that the peak location information is not used. The difference between Figures 13d and 13g is very reasonable, and biasing the distance function by spatial proximity of the texture elements seems like a reasonable way to eliminate the white horizontal streak in the left region.

Example 2 demonstrates the difference when contrast information is used or not in the grouping procedure. Two sets of attributes were considered; one involved peak intensity only and the second included peak contrast. In both cases, spatial location information was used. Figure 14a shows the original TREE-CLOUD sample, and Figures 14b and 14c show the results. Notice that there is a difference between Figures 14b and 14c due to contrast information, and if one considers the original, the difference is not surprising. Due to the "law of similarity," the isolated cloud regions in Figure 14c would group with the tree regions when contrast information is used.

Figures 15 - 20 involve additional texture samples, and the experimental procedures are the same as in the previous examples. The attributes used for all experiments are summarized in Table I.

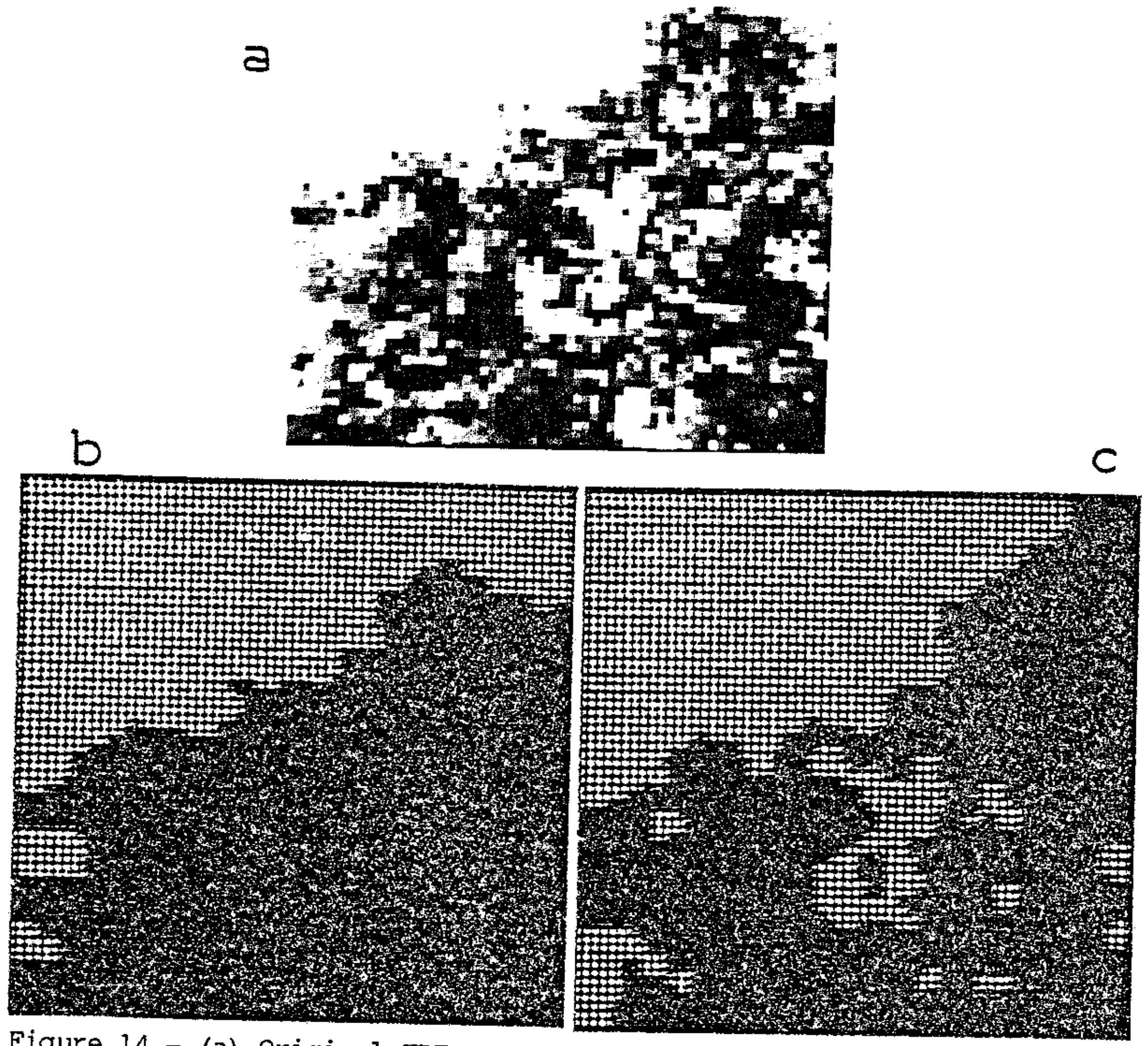


Figure 14 - (a) Original TREE-CLOUD sample, (b) Regions using intensity and location and (c) Regions using intensity, contrast, and location.

The examples in Figures 15 - 17 are trivial in the sense that different textured regions differ in their averaged gray levels. In all three cases, the two significant regions corresponding to two different textures were detected. For instance, in Figure 16, the test sample is a scene composed of a rock against a background of leaves. Obviously the rock has higher intensity, and the regions match well with the original.

Sample	Available Attributes LOC	AH	RH	Threshold
SKY-CLOUD	x	x	x	10
SKY-CLOUD		x	x	25
TREE-CLOUD	x	x	x	8
TREE-CLOUD	x	x		8
BUSH-GRASS (1)	x	x	x	15
LEAF-ROCK	x	x	x	10
TREE-SKY		x	x	6
BUSH-GRASS (2)	x	x		3
LEAF-BRANCH	x	x	x	3
BUSH-WALL	x	x		3

Table I - Attributes used in examples.

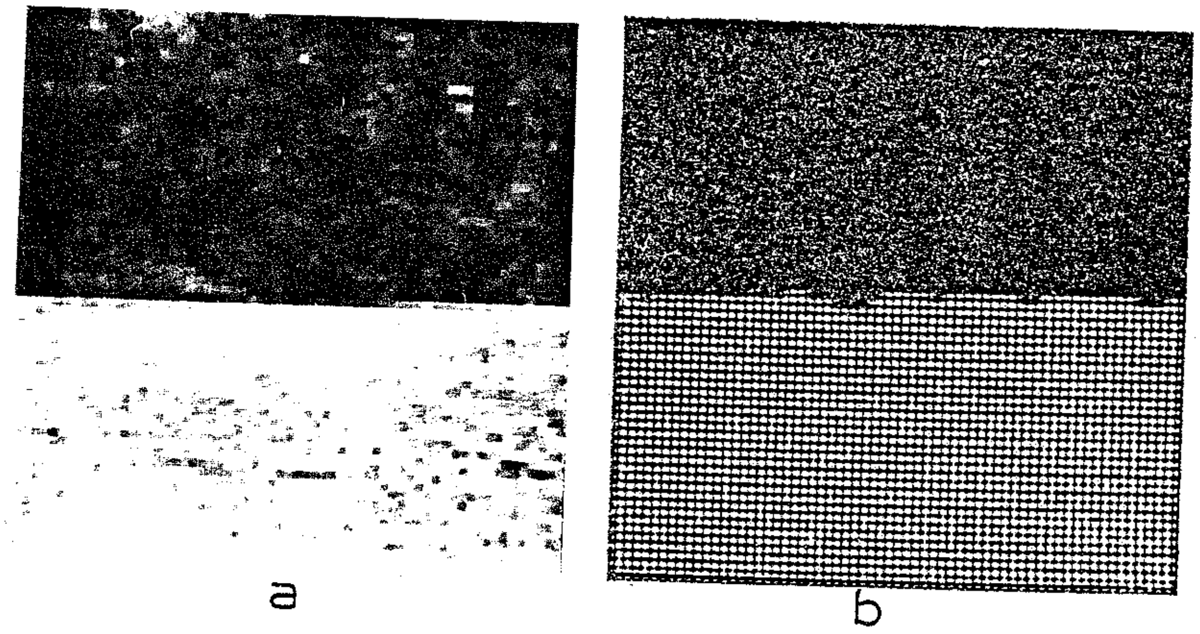


Figure 15 - (a) BUSH-GRASS(1) sample and (b) Regions.

In Figures 18 and 19 the textured images are more complicated than the previous samples; quite a few regions are detected in both samples. It is obvious that only two regions are significant in either sample. The small insignificant regions might be removed or filled out by a smoothing algorithm if it is necessary. Moreover, by comparing with the

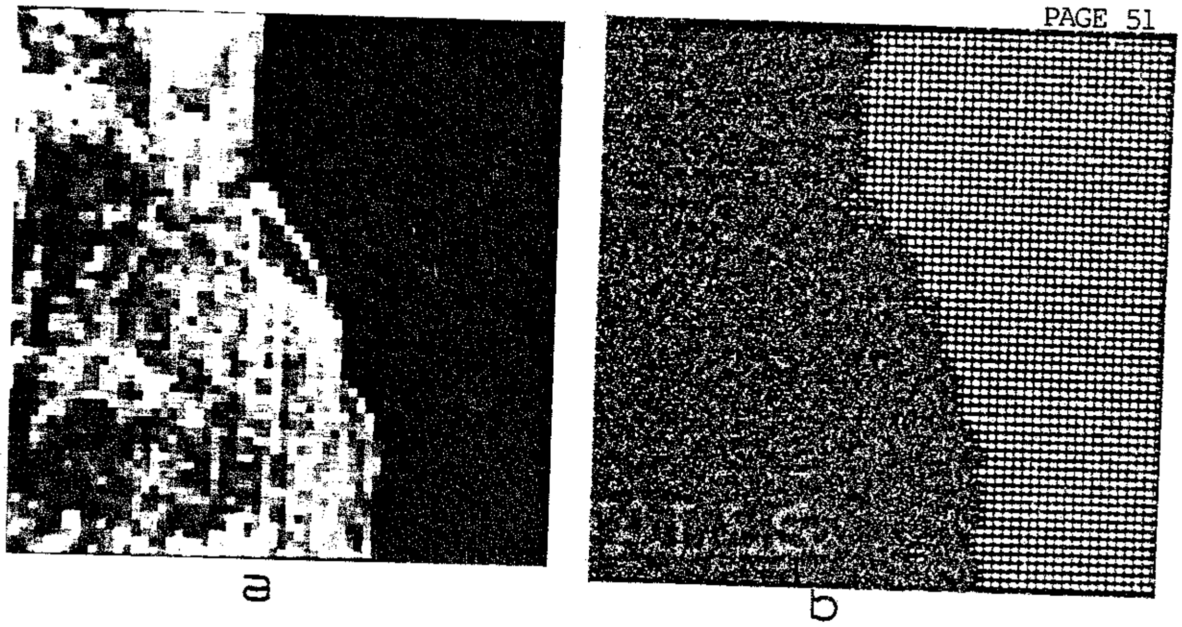


Figure 16 - (a) LEAF-ROCK sample and (b) Regions.

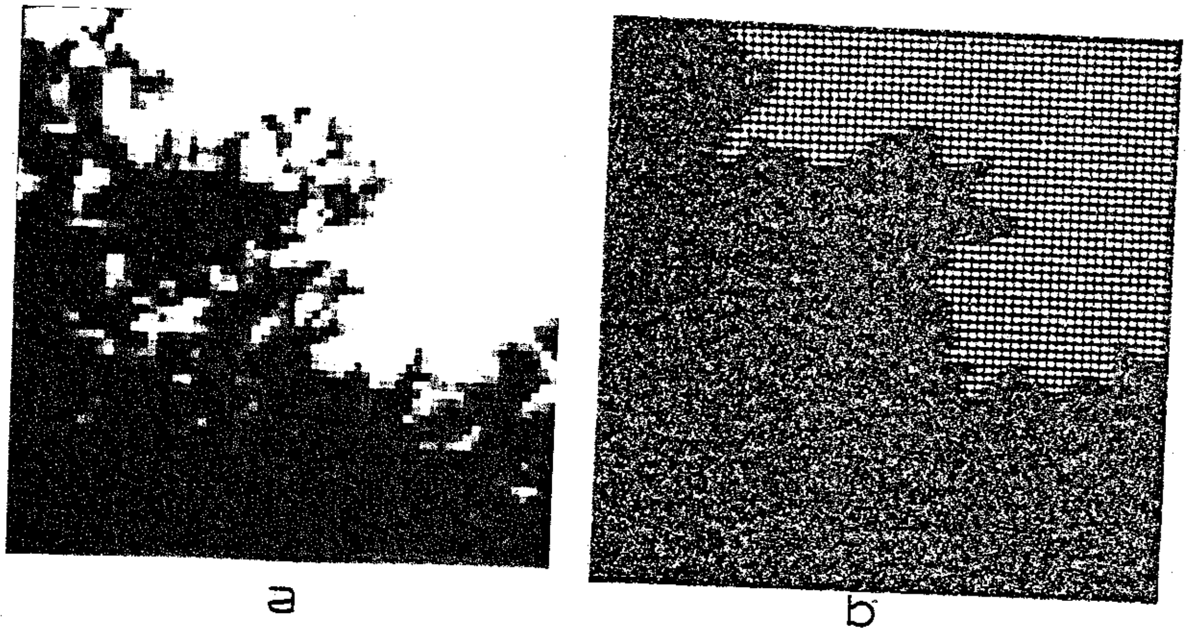


Figure 17 - (a) TREE-SKY sample and (b) Regions.

original images, the appearance of these small regions is not unreasonable. For instance, in Figure 18, the test sample is a scene

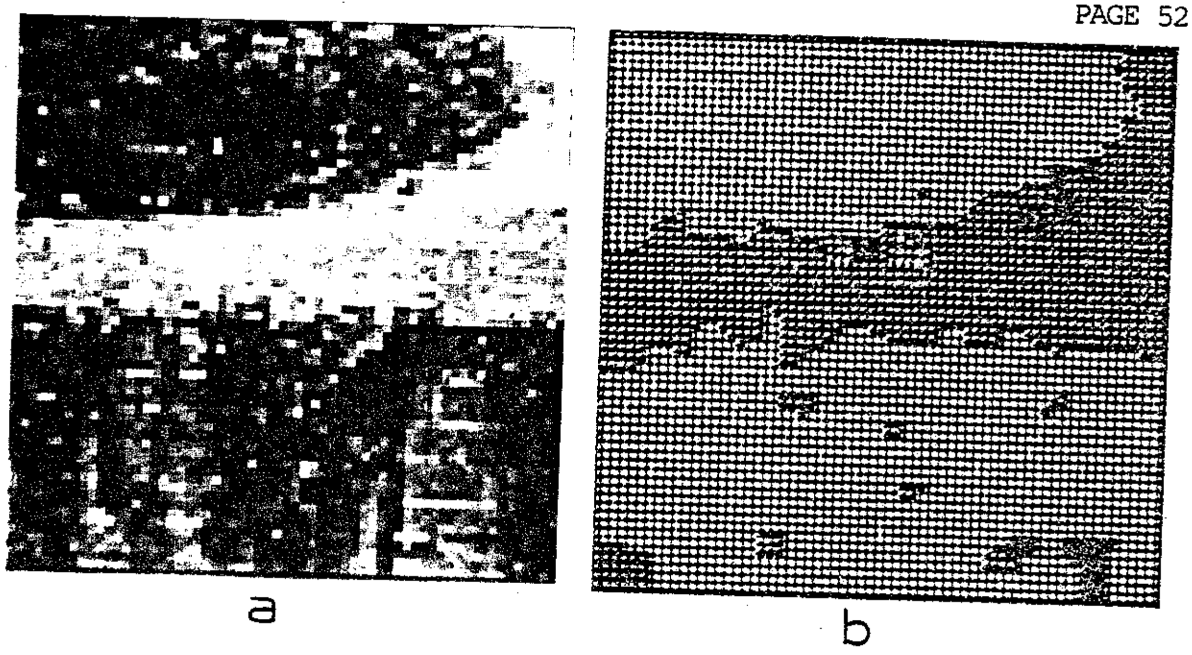


Figure 18 - (a) BUSH-GRASS(2) sample and (b) Regions.

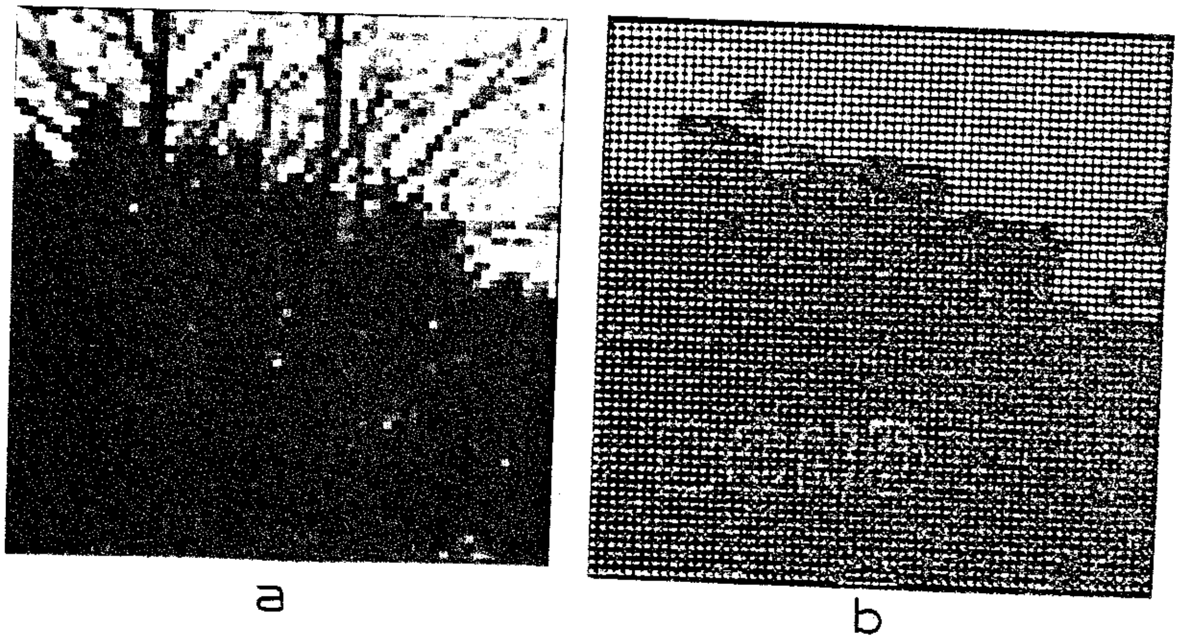


Figure 19 - (a) LEAF-BRANCH sample and (b) Regions.

composed of two bushes separated by the grass field as background. The two bushes are assigned the same label in most parts, and the grass

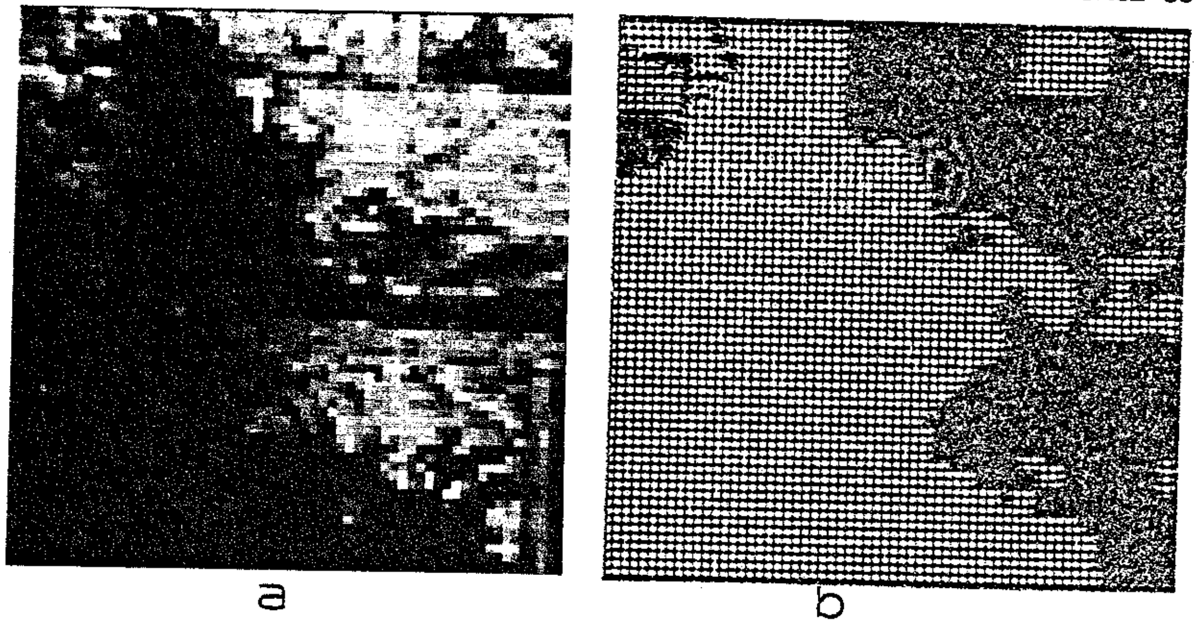


Figure 20 - (a) BUSH-WALL sample and (b) Regions.

field is assigned a different label. One might expect a few small regions with grass labels to appear in the lower parts of the bushes; also, small regions with still other labels might be expected at the boundary between bushes and grass.

Figure 20 is the most difficult one among the samples tested. In this sample, the scene is composed of a bush against a heavily textured wall. Due to insufficient resolution, there are a few portions of the wall which resemble the bush in features such as average intensity. Therefore, these portions are grouped with bush, but the results are still satisfactory.

In the experiments, the results of applying a MST technique for texture region growing based upon structural adjacency relationships are very encouraging. The number of elements extracted is only about 1/10 of the number of elements in the image array. Moreover, one may speculate that additional attributes such as orientation and shape would produce additional improvements. Finally, the results shown here are based only upon the pseudo-2D primitives obtained by the one-dimensional peak detection operator.

5.4 Results Using the ART

In this section we describe experiments in which texture elements were extracted using the ART, rather than the local peak extractor. The asymptotic relational tree algorithm described in Section 4.2 does not represent peak structure directly. It describes only the slice-to-slice relationships between the intensity sliced regions of the picture function. In order to extract peak structure, a consolidating procedure is required. This consolidation procedure traverses the ART along all paths from frontier to root. Whenever a node is found that has no brothers, it is merged with its parent node as shown in Figure 21. This prevents a high contrast texture element with no substructure from being repeatedly sliced, and it eliminates from the ART redundant vertices.

The consolidation procedure provides the technique for extracting the peak structures that are used as the texture elements. The next step is to measure the attributes from these X-Y-INTENSITY blobs. The attributes computed from these 3D blobs are called 3D attributes. At the same time that the tree nodes are consolidated, information about each peak's relational structure and its 3D attributes can be computed and stored.

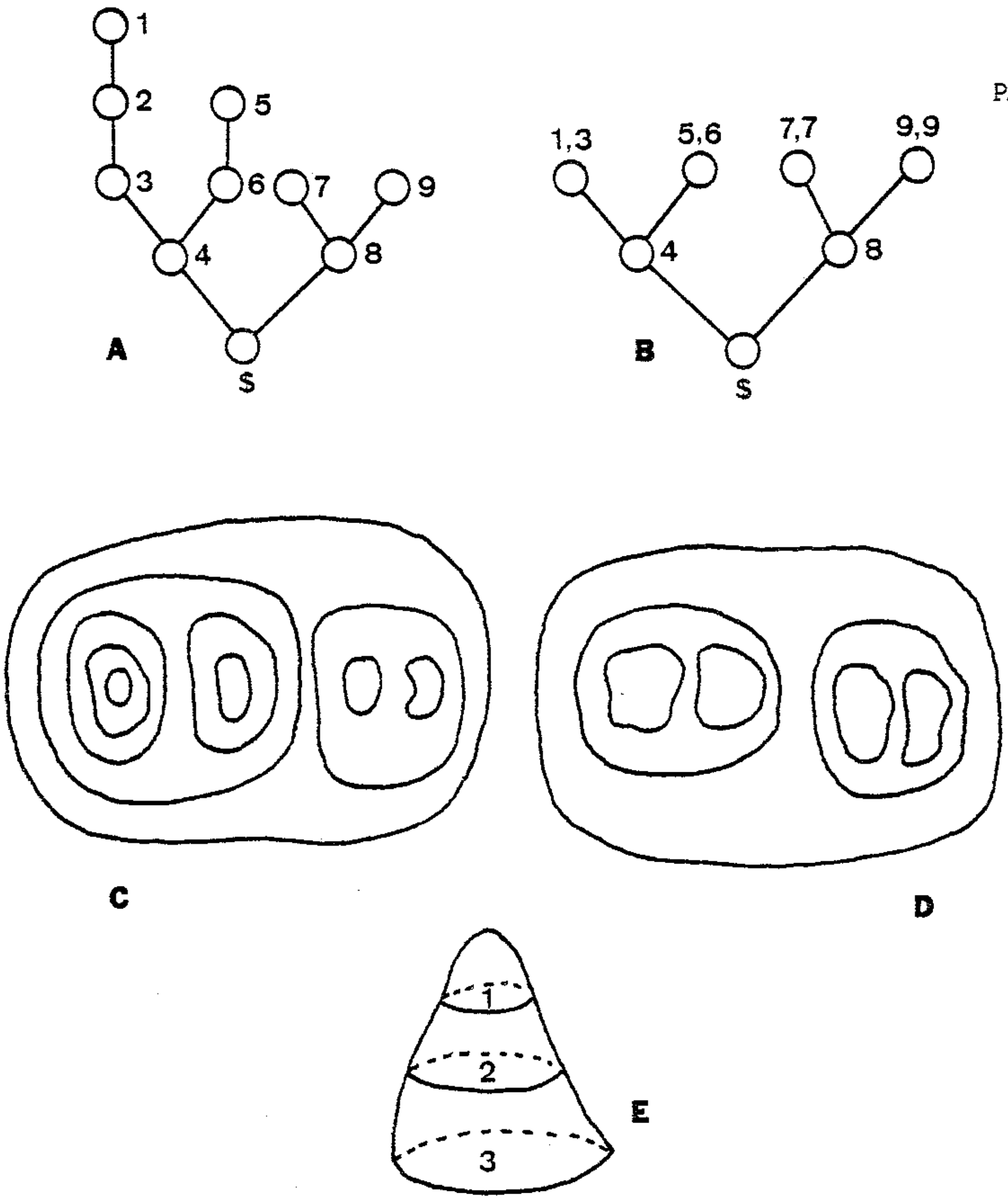


Figure 21 - (a) An ART, (b) The consolidated ART, (c) Contour graph of (a), (d) Contour graph of (b), and (e) 3D peak structure.

Experiments were run on the same 8 texture samples as in Section 5.3. The available attributes are illustrated in Figure 22, and the corresponding primitive distance functions are

(1) base center: $f_{BC}(i,j) = |X(i)-X(j)| + |Y(i)-Y(j)|$

(2) absolute height: $f_{AH}(i,j) = (AH(i)-AH(j))^2$

(3) relative height: $f_{RH}(i,j) = (RH(i)-RH(j))^2$

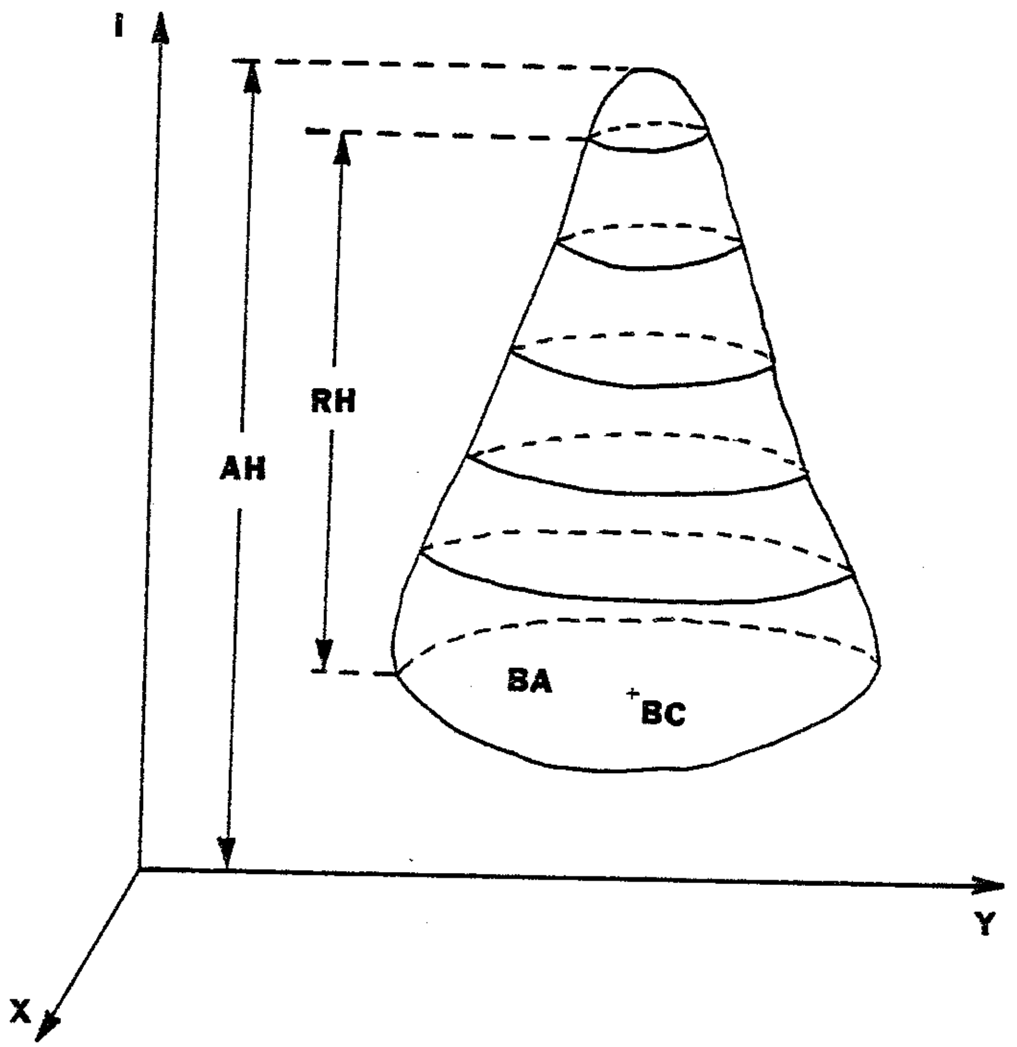


Figure 22 - Illustration of attributes measured from a 3D X-Y-INTENSITY blob.

As before, the edge weights in the texture graph were computed by using distance functions that were sums of some of the primitive distance functions given above.

All the original textures have 256 available gray levels, and the number of slices used for the construction of the ART's is 16. Figure 23 shows the ART and the consolidated ART for the TREE-CLOUD texture. Table II shows the attributes that were used for the distance function for each of the 8 texture samples.

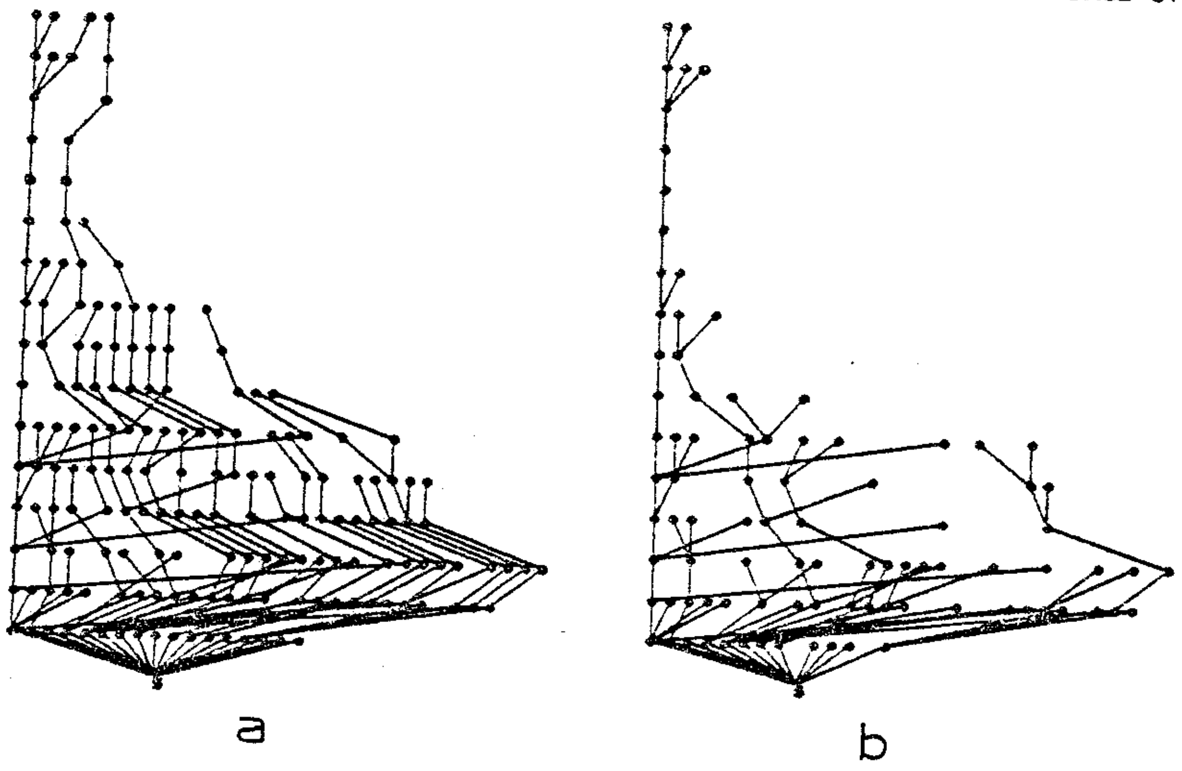


Figure 23 - (a) ART and (b) Consolidated ART for TREE-CLOUD texture.

	Available Attributes			
Sample	LOC	AH	RH	Threshold
SKY-CLOUD	x	x	x	5
TREE-CLOUD	x	x	x	3
BUSH-GRASS (1)	x	x		10
LEAF-ROCK	x	x	x	25
TREE_SKY	x	x	x	3
BUSH-GRASS (2)		x		6
LEAF-BRANCH	x	x		2
BUSH-WALL	x	x	x	4

Table II - Attributes used in examples.

The results after application of the MST technique on the extracted 3D blobs are shown in Figures 24-31.

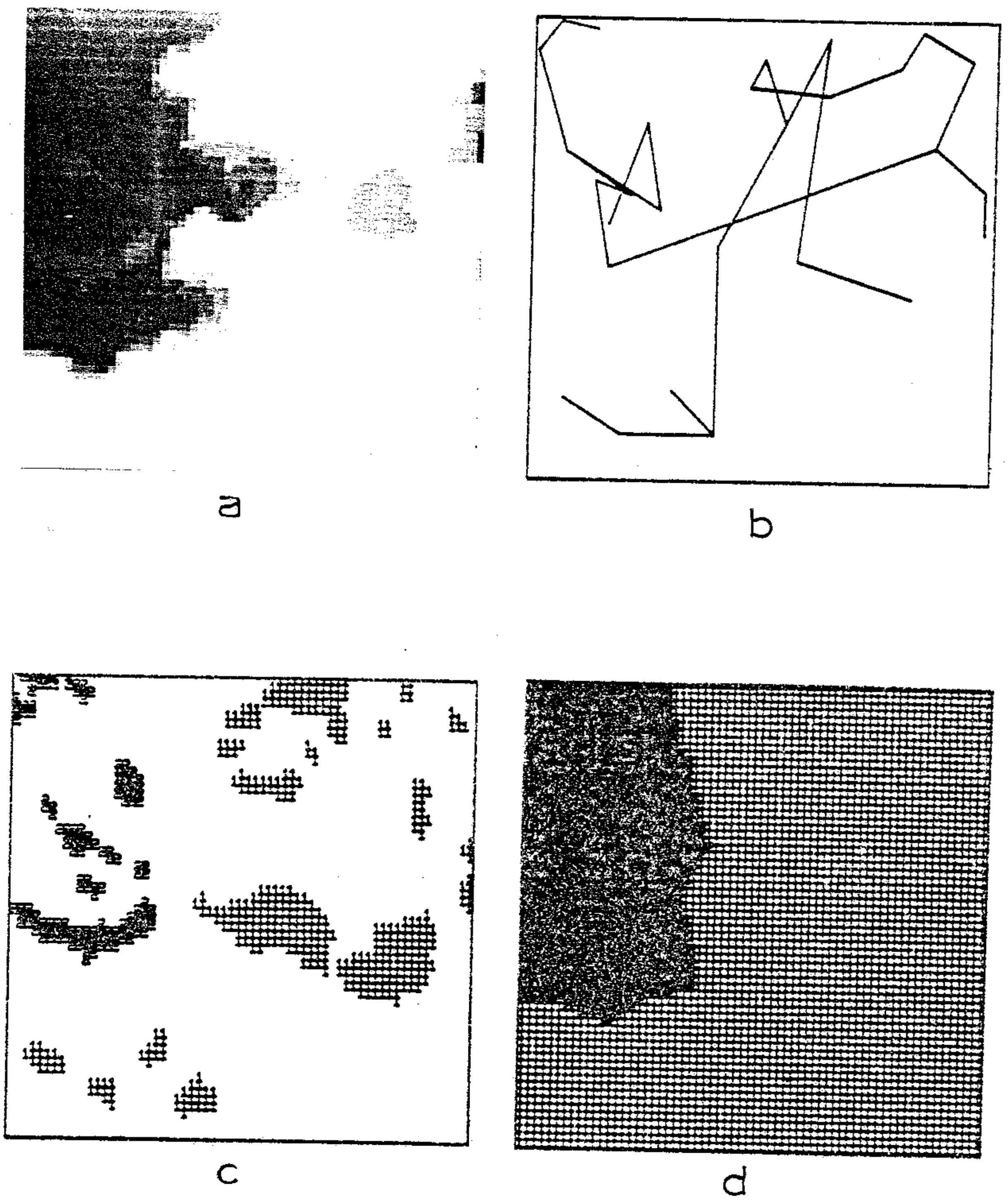


Figure 24 - (a) Reconstructed SKY-CLOUD, (b) MST, (c) Labeled blobs, and (d) Regions.

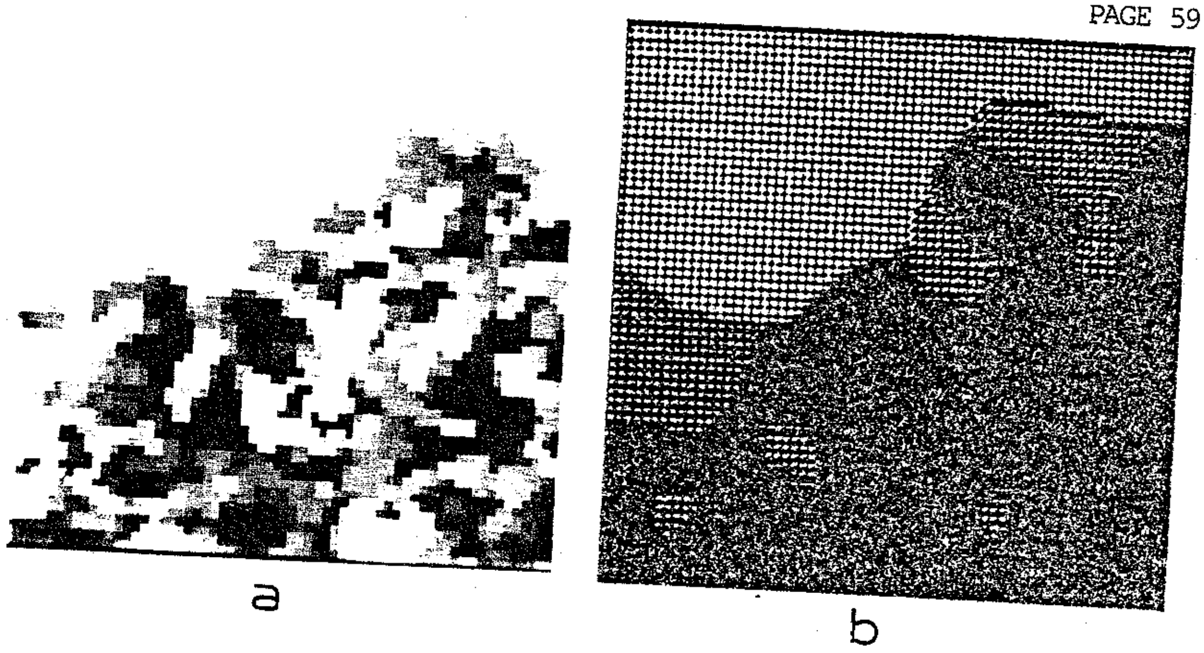


Figure 25 - (a) Reconstructed TREE-CLOUD and (b) Regions.

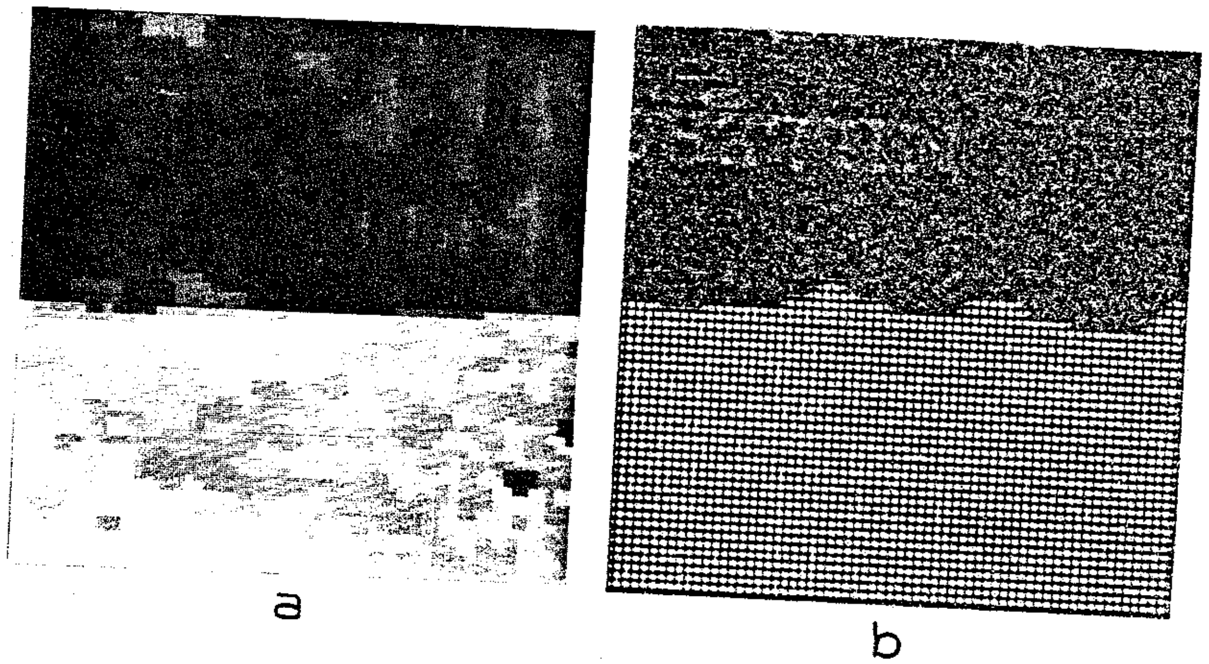


Figure 26 - (a) Reconstructed BUSH-GRASS(1) and (b) Regions.

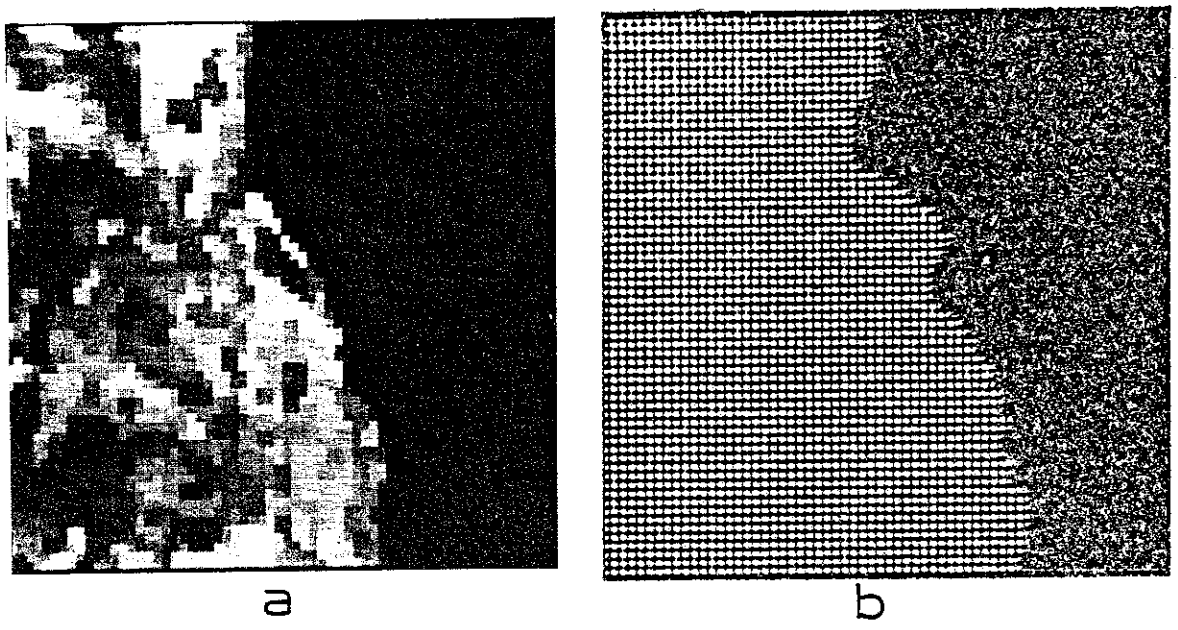


Figure 27 - (a) Reconstructed LEAF-ROCK and (b) Regions.

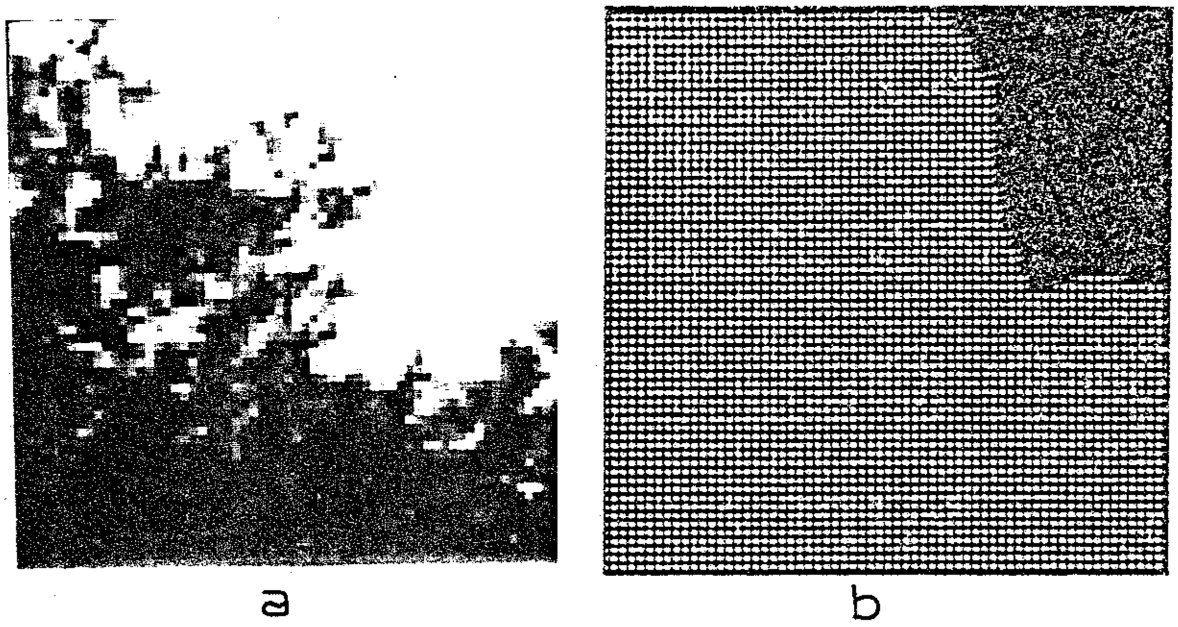


Figure 28 (a) Reconstructed TREE-SKY and (b) Regions.

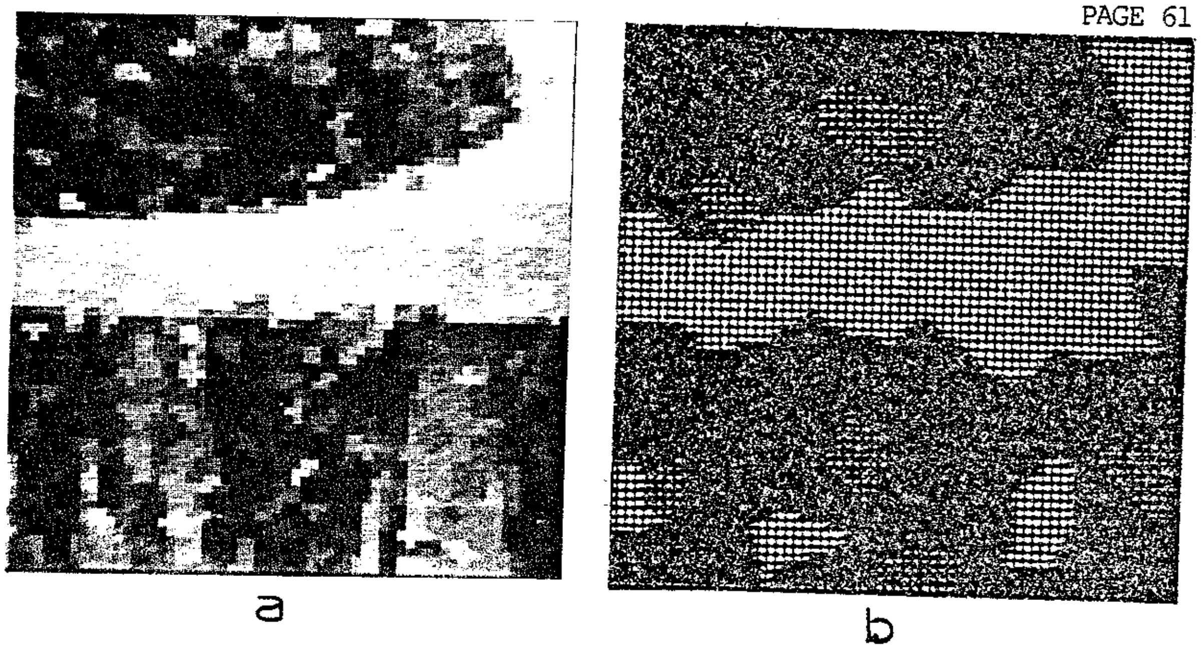


Figure 29 - (a) Reconstructed BUSH-GRASS(2) and (b) Regions.

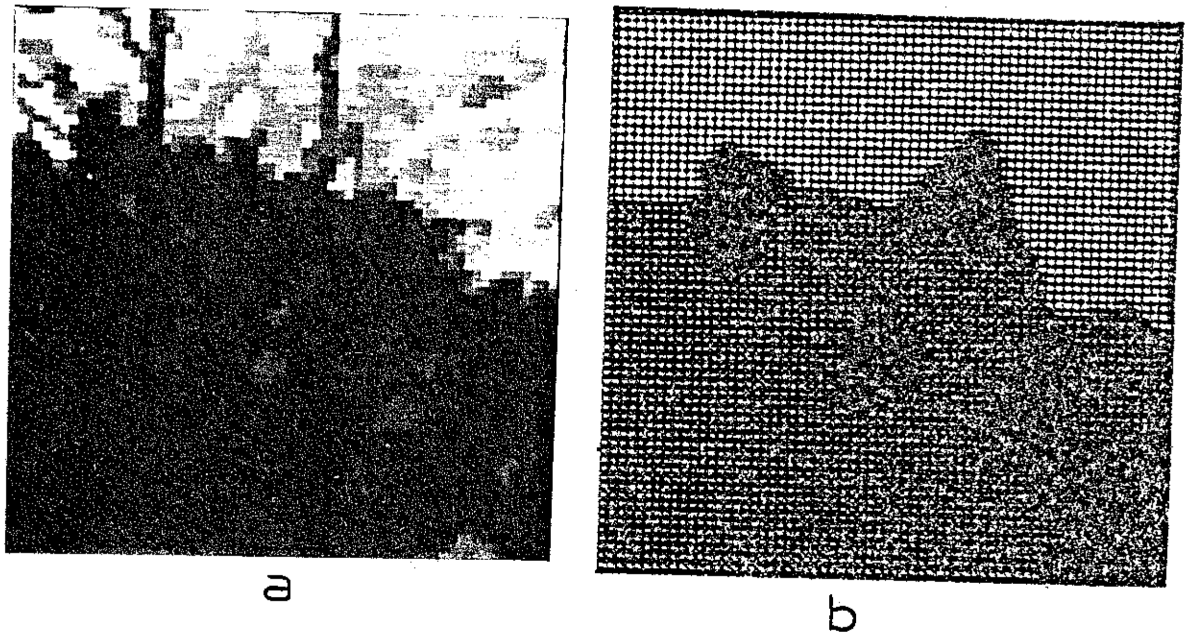


Figure 30 - (a) Reconstructed LEAF-BRANCH and (b) Regions.

One result that is very striking is that the MST's constructed are far simpler than those for the previous results due to a tremendous

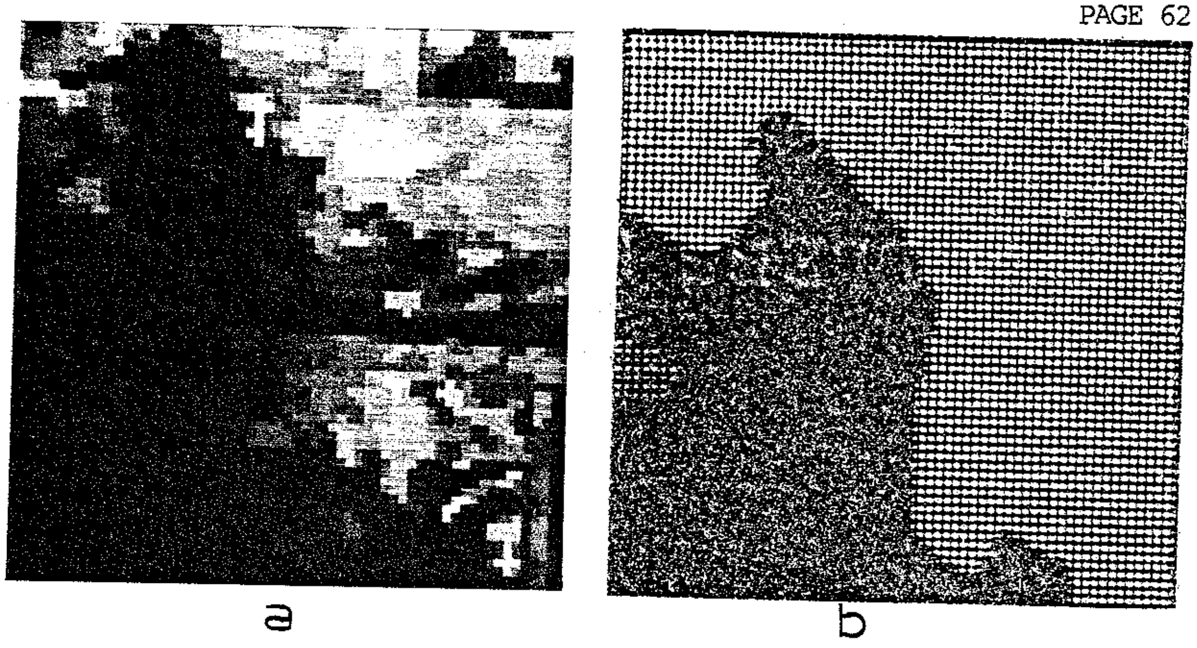


Figure 31 - (a) Reconstructed BUSH-WALL and (b) Regions.

reduction in the number of texture elements. For instance, in Figure 24, only 28 blobs were extracted from the corresponding ART. The result after application of the diffusion algorithm is shown in Figure 24d. If one compares the regions obtained with the original, the boundary is acceptable even though it is less precise due to the small number of texture elements that were extracted.

The results shown in Figures 26 and 27 are satisfactory even after the application of the diffusion algorithm due to the fact that a sufficient number of blobs were extracted which were evenly spread out over the entire image space. The situation in Figure 28 is the same as in Figure 24 in the sense that an insufficient number of blobs were extracted for good results from the diffusion algorithm.

Compared to the experimental results using the local-peak extractor, the results in Figures 29-31 are also satisfactory, but the appearances of the boundaries are not so good. This is due to the small number of texture elements and also to the fact that only 16 slices were used in constructing the ART.

6. Region Growing by Tree Pruning

In this section a texture region growing technique is explored that functions by making use of the spatial adjacency relationships among texture elements. The basic idea is to merge microtexture elements at the frontiers of the asymptotic relational tree until texture elements are eliminated from the regions that contain them. The region grower consists of the following steps:

(1) Constructing the asymptotic relational tree (ART)

(2) Pruning frontier vertices from the relational tree structure

(3) Reconstructing the texture from the pruned tree

Steps (2) and (3) are iterated until a level is reached at which global or macro regions appear.

Reconstruction of a texture from the ART is done quite easily by assigning the average gray level of a texture element to the points which it contains. Other attributes such as area and shape can also be used for the reconstruction.

Figure 32 gives an example of the reconstrction of a texture from the relational data structure. The original textue sample (RUG) is shown in Figure 32a, and the reconstructed texture is shown in Figure 32b. Notice that the essential features of the texture are preserved while the extremely complex but irrelevant details have been removed.

Pruning frontier vertices from the relational tree is a natural way to group peak substructures; under the assumption that macrotextures are

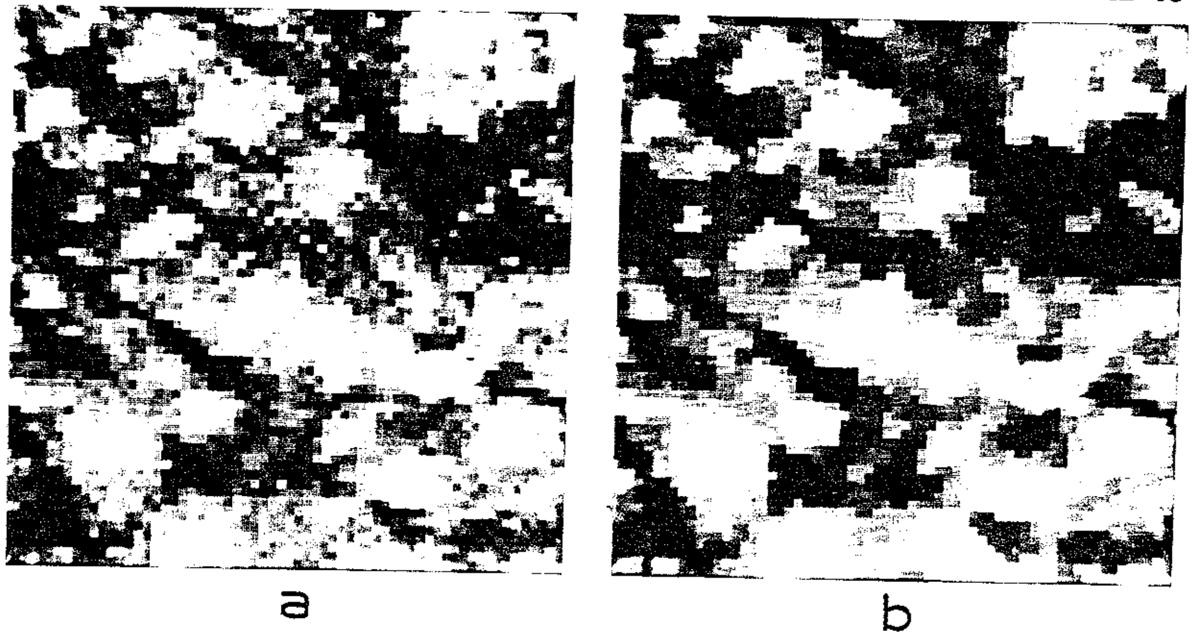


Figure 32 - (a) Original RUG sample and (b) Reconstruction from 8-level ART.

those elements which are composed of spatially adjacent microelements, images with microtextures can be transformed into macrotextures by utilizing the pruning procedure. Pruning is an iterative procedure; first the ART is constructed for a texture image, and then pruning is done for several iterations. Next the texture is reconstructed from the pruned tree structure. The reconstructed texture will contain larger texture elements that reflect the merged microtexture elements.

Removing frontier vertices whose size is insignificant has the effect of two-dimensional smoothing when the texture is reconstructed. The size of the texture element can be determined by computig the area of the corresponding region or the volume of the corresponding peak structure.

Pruning based upon the spatial adjacency relationships among texture elements is closely related to proximity grouping in the human visual system since pruning has the effect of merging spatially adjacent elements together. The spatial adjacency relationships are hierarchically represented in the relational tree structure. Two simple rules have been devised for doing the tree pruning.

6.1 Pruning Techniques

Rule P: (Pruning only)

(1) For each frontier vertex check if it has any non-frontier brother.

  (1.1) If YES, check NEXT frontier vertex.

  (1.2) If NO, prune the frontier vertex.

(2) After all frontier vertices have been scanned, mark the associated father vertices frontier.

(3) Reconstruct the texture from the updated tree structure.

(4) Return to step (1).

Rule G (Pruning and Grouping)

Besides pruning, rule G attempts to group together frontier vertices which came from the same parent vertex even if they have one or more non-frontier brothers. The updated information about grouped frontier vertices is stored in the eldest brother. The procedure is implemented by modifying step (1.1) of Rule P in the following way:

(1.1) If YES, group the frontier vertex with the eldest brother.

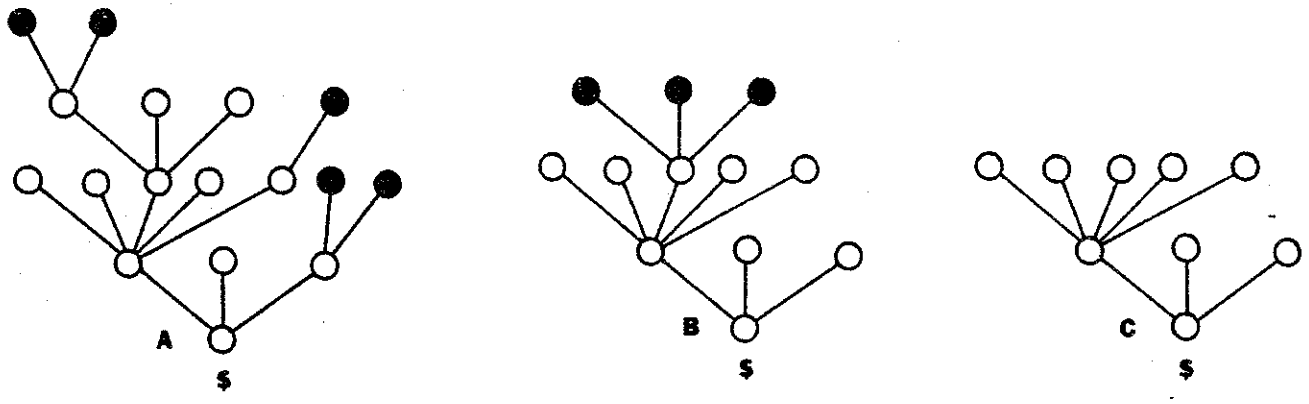


Figure 33 - Iterative pruning using Rule P; (a) Original, (b) First iteration, and (c) Second iteration.

Rule G has the remarkable effect of grouping those subregions that would not have been merged until many iterations were performed if only rule P were applied. Figure 34 shows the corresponding trees of the step-by-step iteration.

In our experiments with proximity pruning, the only information used was the spatial information in the sense that in most cases the regions are expanded according to the hierarchical containment relationships embedded in the relational tree. However, in specific applications where semantic information is available, pruning can be guided in a more intelligent way. For example, information such as shape and size of the texture elements can be used as constraints for pruning.

6.2 Examples

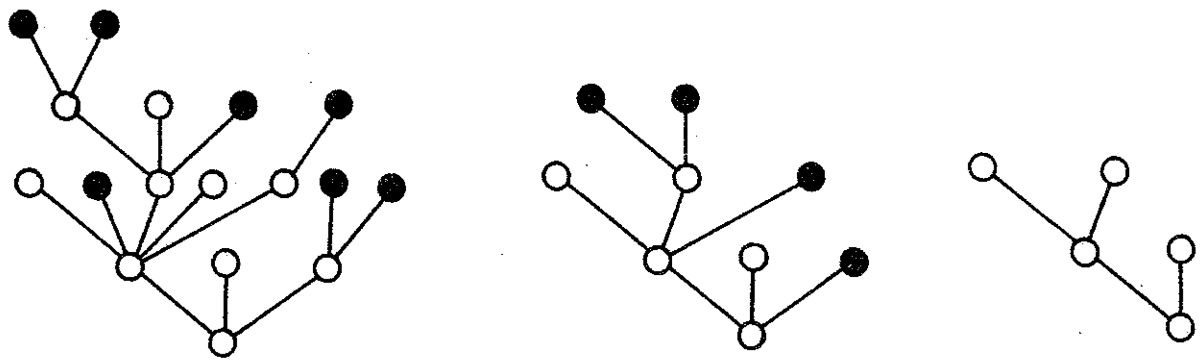


Figure 34 - Iterative pruning using Rule G; (a) Original, (b) First iteration, and (c) Second iteration.

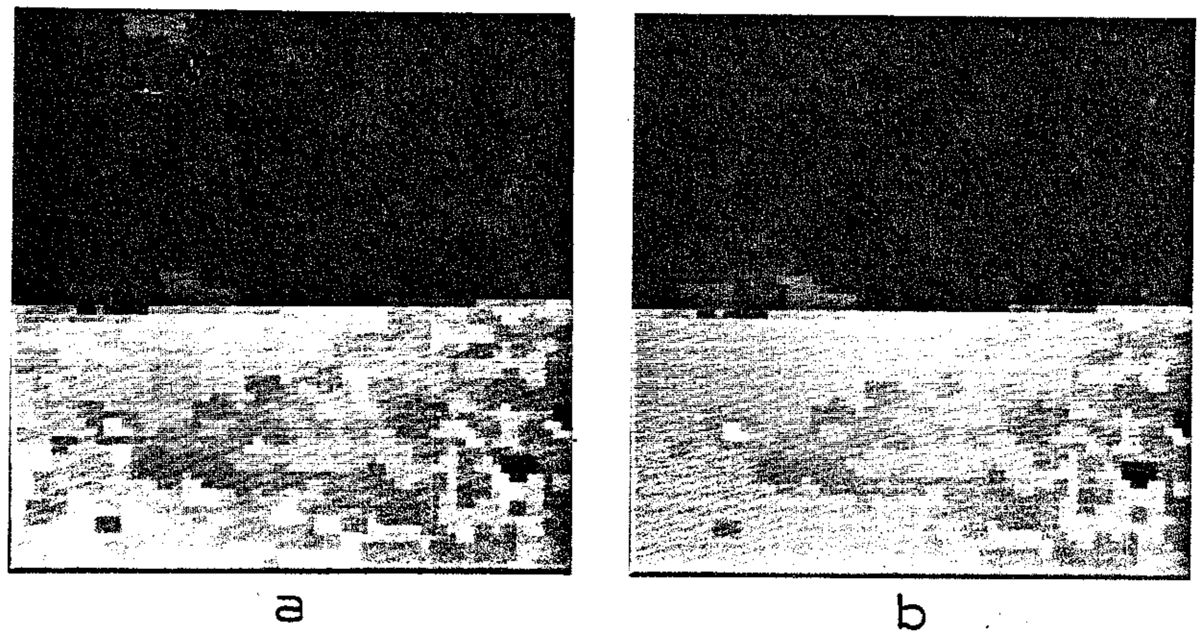


Figure 35 - (a) Reconstructed BUSH-GRASS(1) and (b) Pruning after 3 iterations.

Shown in Figures 35 through 39 are examples of pruning experiments

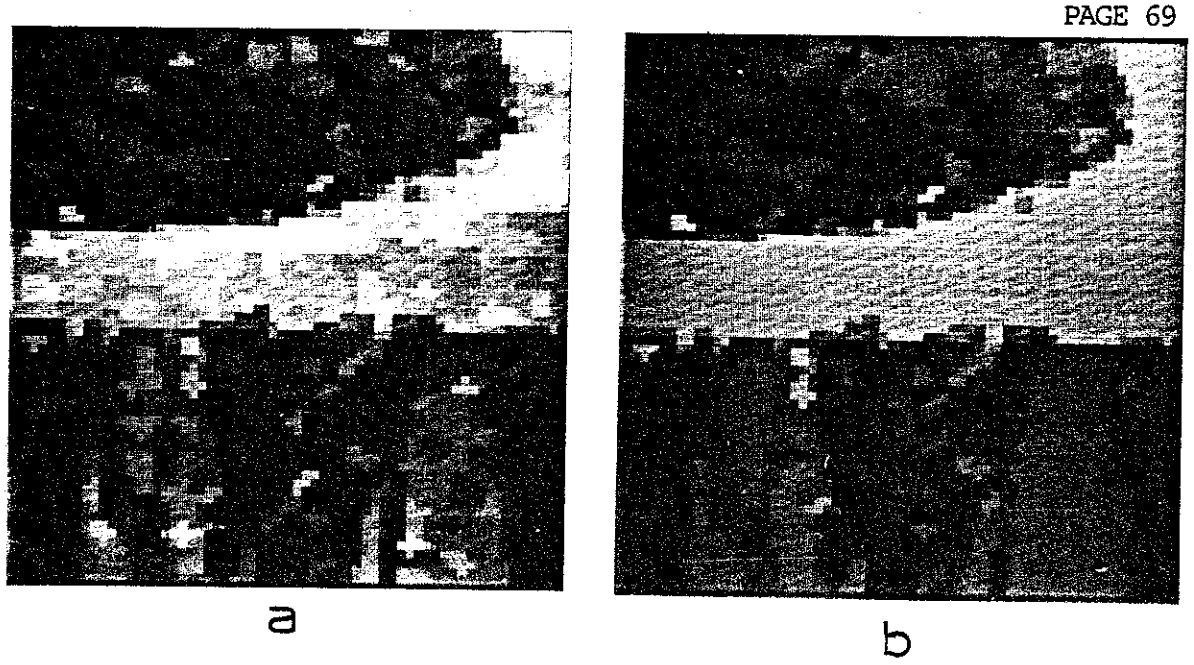


Figure 36 - (a) Reconstructed BUSH-GRASS(2) and (b) Pruning after 3 iterations.

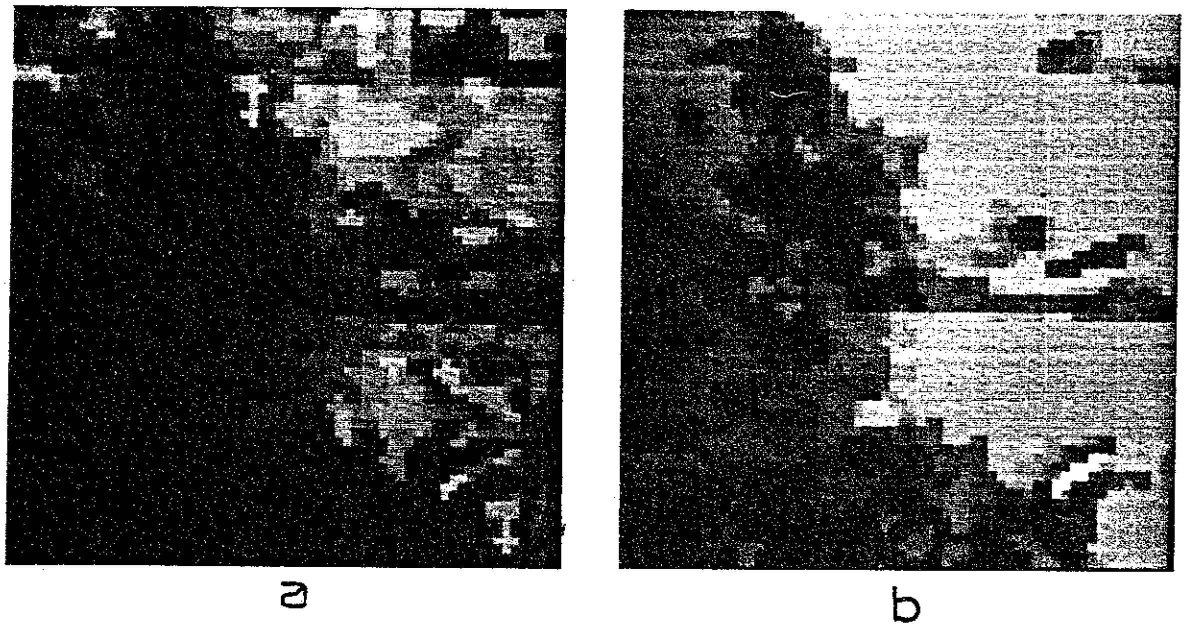


Figure 37 - (a) Reconstructed BUSH-WALL and (b) Pruning after 5 iterations.

using Rule P. The original samples have size 64 x 64 and 256 available gray levels. In all examples, the number of slices used for the

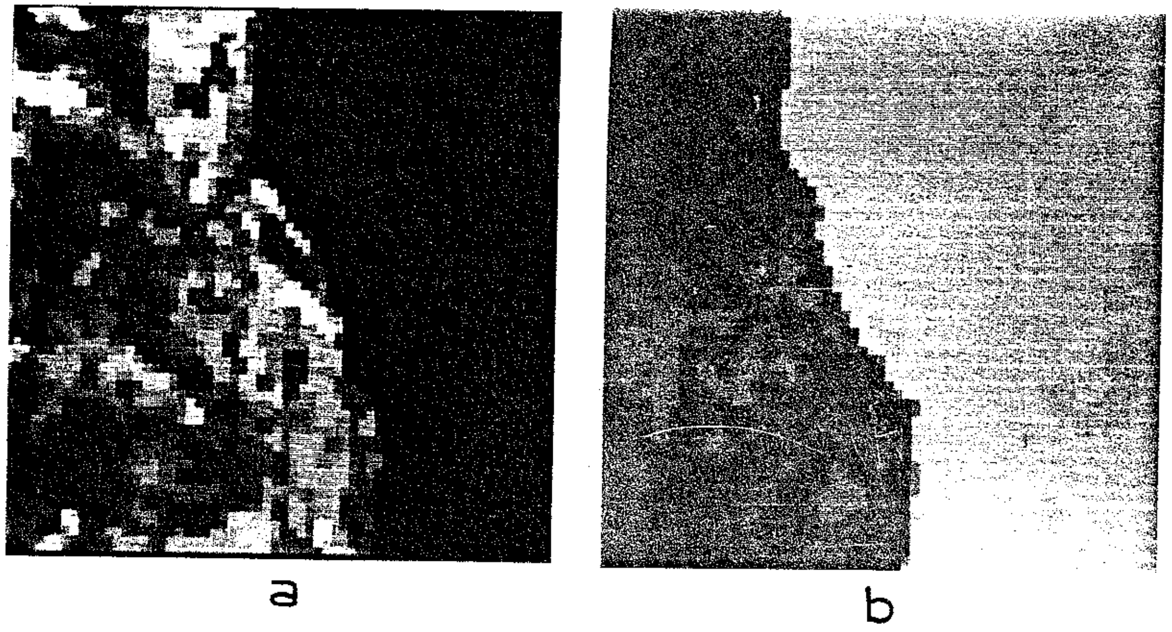


Figure 38 - (a) Reconstructed LEAF-ROCK and (b) Pruning after 7 iterations.

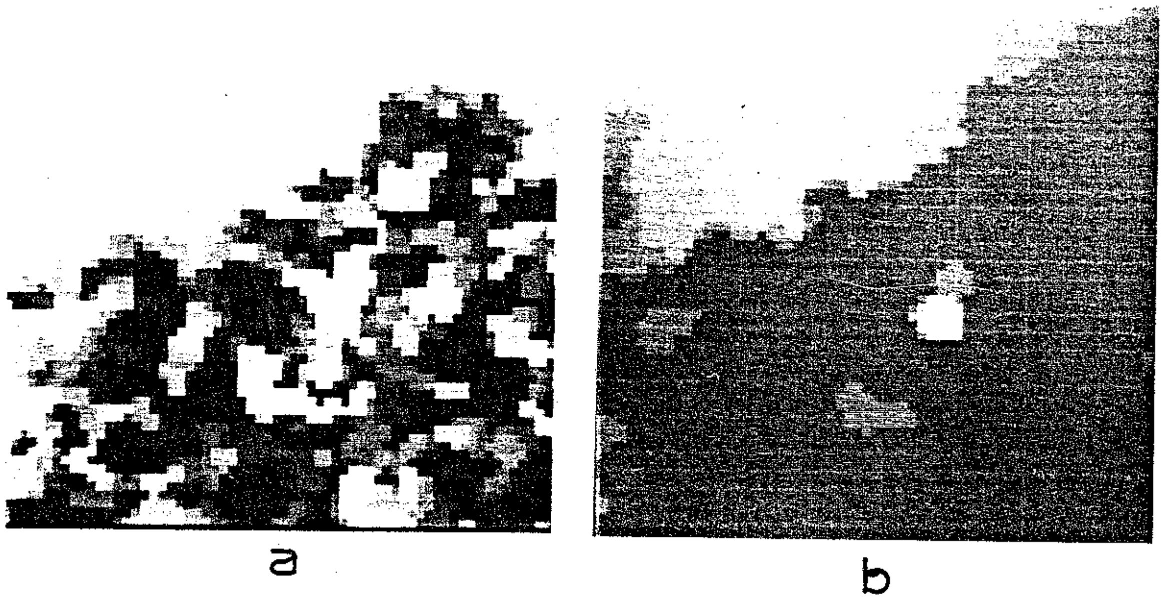


Figure 39 - (a) Reconstructed TREE-CLOUD and (b) Pruning after 9 iterations.

construction of the ART is 16. Experiments using Rule G are not shown because the results are not significantly different.

7. Discussion

The two region growing procedures suggested in this paper are based upon a structural model of texture. It is assumed that texturė can be described by the texture elements which are three-dimensional X-Y-INTENSITY blobs; each such blob is characterized by a list of attributes. The proposed structural model attemps to provide a framework for describing these texture elements from which a texture is constructed and the aggregation mechanisms by which these elements form Gestalts. Then the boundary detection problem is approached by looking for changes either in the elements themselves or in the relationships between them. The structural approach transforms a difficult statistical problem into a difficult structural problem, but we believe it is more feasible to deal with the pieces of the structural problem because the computational and the psychological issues are not all mixed in together. Both procedures also make use of Gestalt grouping principles. The procedures are region growing approaches rather than edge detection approaches, and no differentiation or other preprocessing of the image is required. Compared to the conventional region growing techniques, the region growers provide the following advantages.

(1) The regions growers make explicit use of Gestalt principles such as similarity and proximity.

(2) The consequences of applying Gestalt principles leads to region growers that use a more global approach.

(3) The relational tree and the spatial data structure provide a dependable and efficient data structure for experimentation.

7.1 Region Growing Using the MST

The MST based region grower is attractive because proximity and similarity grouping of texture elements can be very explicitly controlled by specifying the weights on the edges of the texture graph. The MST itself connects the texture elements together in such a way that each element is connected to its closest neighbor. Even though constructing the MST requires $O(n^2)$ computations, since we are dealing with macroelements, n is small. Furthermore, there is no reason why approximate computations could not be used to construct the MST, such as Hall's [Ha73], which is $O(n^{1.33})$.

There are a number of other experiments that still need to be done with the MST. For example, since the regions one obtains depend upon which edges are broken in the MST, it might be worthwhile considering other definitions of inconsistency or increasing the neighborhood over which inconsistency is computed. We have not yet found a good algorithm for setting the inconsistency threshold automatically. The number of regions labels one obtains depends upon this threshold. Another important problem involves the definition of the distance functions by means of which proximity and similarity are defined. The entire grouping procedure can be run by using a single distance function as we have done in our experiments, or grouping can be done independently for each attribute. In Figure 40, for example, grouping according to similarity would produce three region labels. If grouping according to spatial proximity were to be done next on the small circles alone, two distinct regions would be produced -- small circles left and small circles right -- so that four regions would be produced in all.

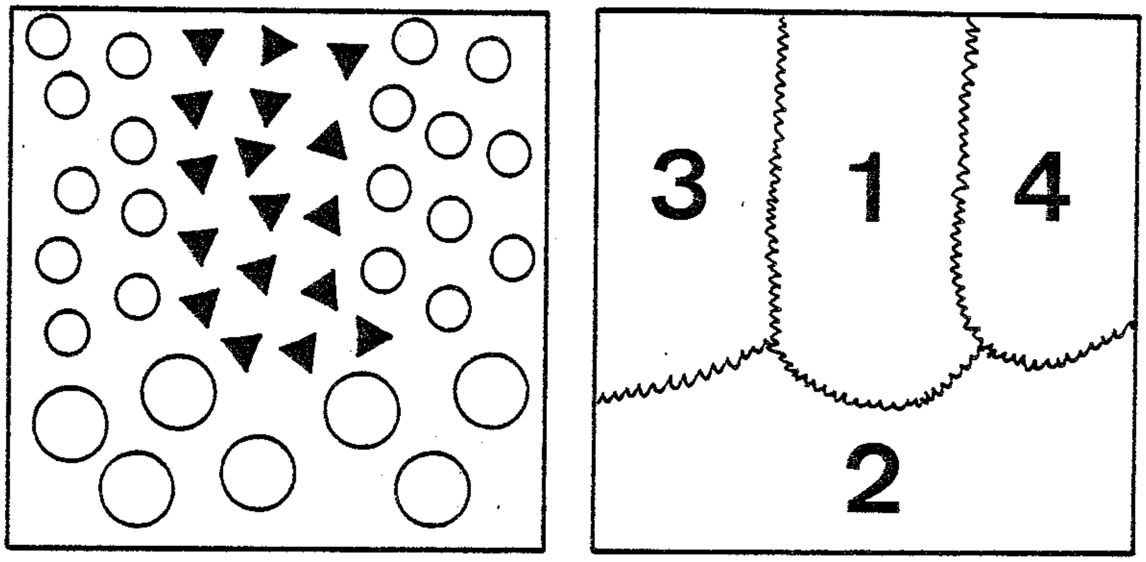


Figure 40 - Texture sample with three element types and four regions.

7.2 Region Growing by Tree Pruning

The pruning based region grower is attractive because the computations are simple once the R-tree has been constructed. Frontier vertices of the tree with close common ancestry are spatially close, and the tree can be searched vertically to check the similarity of microtexture elements whose merger is pending. However, it is not necessarily true that spatially close elements are also closely related in the R-tree since two low contrast elements might be separated by a deep valley. Thus, at the same time one achieves simplicity, one gives up some of the freedom to elect grouping criterea.

If one compares the results in Section 6 with those of Section 5, it appears that as a region grower, pruning is less successful than MST techniques on the same images. However, pruning is a fairly useful technique for producing macrotexture elements from microtexure elements, and pruning might be successfully combined with the MST region grower.

7.3 R-trees

Clustering by MST or by tree pruning relies upon the extraction of texture elements based upon the relationships among the light and dark regions of a texture. Of course, since no prior world knowledge is used in extracting these texture elements, if such knowledge is available, R-trees will not be a sufficient technique.

One other major consideration in the use of R-trees is figure-ground. R-trees are not symmetric in the sense that the same R-tree will not represent simultaneously white primitives on black background and

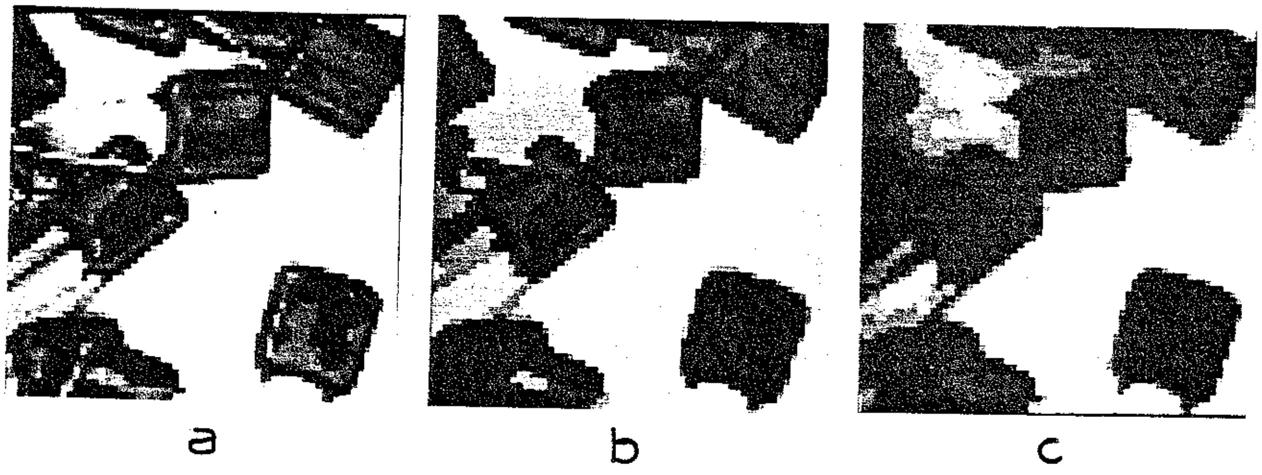


Figure 41 - (a) Original CAPS scene, (b) Pruning after 5 iterations, and (c) Pruning the negative image after 5 iterations.

black primitives on white background. Figure 41 shows the effect of pruning on a capacitor scene by merging white elements (41b) and by merging black elements (41c). Since the capacitors are black objects with white spots, white macrotexture elements are produced in Figure 41b, whereas in Figure 41c, the black capacitors "consume" the microtexture elements to produce excellent silhouettes. Five pruning iterations were done in both cases.

The results obtained by using the ART were slightly worse than those obtained by using the local peak extractor. As noted, this is attributed to the small number of levels in the ART and to the small number of primitives that were extracted. The problem of having a very small texture sample (or one with large primitives) may be a drawback for any procedure, and since our software was limited to 64 x 64 samples, we also encountered such problems. A sample called MIX was generated by taking one quadrant from each of four different but similar textures from the Brodatz collection [Br66]. The original image is

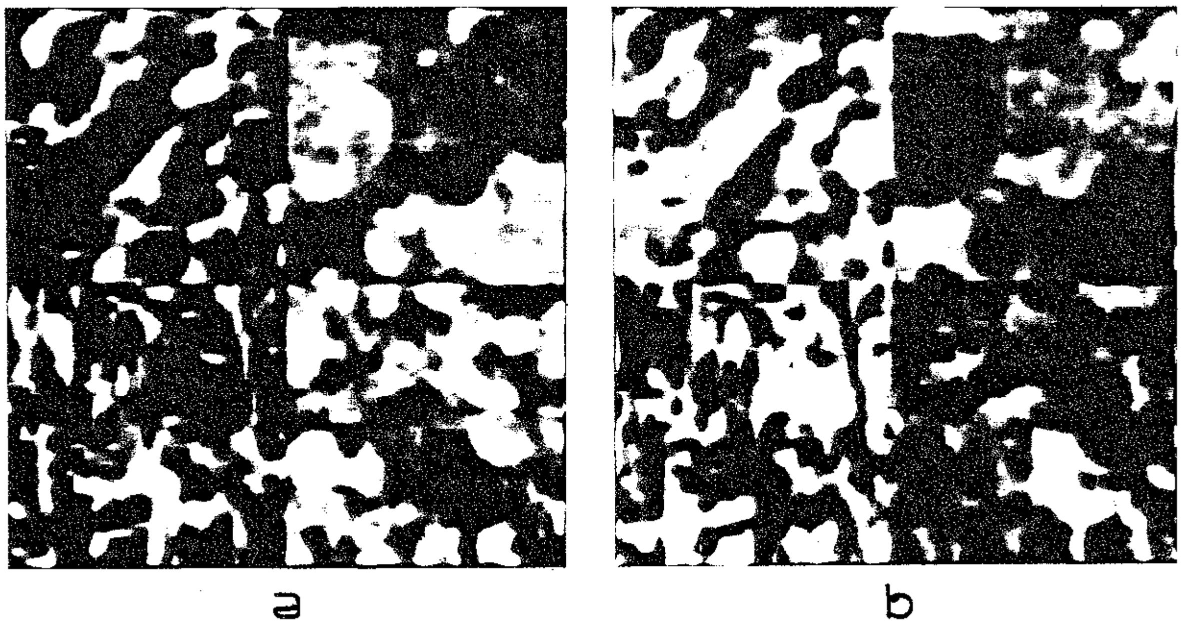


Figure 42 - (a) MIX texture collage and (b) Negative.

shown in Figure 42a and the negative is shown in Figure 42b. By looking at the four quadrants in Figure 42a, one notices that the upper left one shares shape similarity with the lower left one, but they differ in orientation. The upper right quadrant shares similarity with the lower right, but they differ in size. No matter how the distance function was constructed, the MST procedure was not able to separate the top and bottom halves satisfactorily. One of the problems is that due to insufficient resolution, global information about textured regions is lost during the construction of the ART structure. In addition, important information required for the discrimination of the four quadrants are size and shape; these were not computed due to computer memory limitations.

7.4 Further Investigation

The following are considered important areas for further investigation.

(1) More attributes

As mentioned above, it would be good to include shape measures such as base area and peak volume. No 3D shape measures have been tested to date. The underlying principle of the current work has been to keep the attribute measurements as simple as possible, since the use of complicated measures would have required greater computational resources than we had available.

(2) Figure-ground

The CAPS example illustrates the importance of the figure-ground problem. Peaks and valleys are dual structures, though they are not treated symmetrically. If figure-ground issues are considered to be resolved at the time when region labels are attached, then it would be desirable to keep simultaneously the competing figure-ground assertions. In that case one would have a data structure that contains not only the relational tree of the original image, but the tree for the negative as well.

(3) Deep structure

In our work to date it has been assumed that all grouping of texture elements in the R-tree involves only the vertices at the frontier of the tree. However, it is reasonable to assume that on the basis of similarity, merging regions that correspond with internal vetices of the R-tree would produce even better results. The MST might provide an effective technique for grouping regions in the deep structure. Suppose all internal tree vertices are treated as representations of completely independent texture elements; in forming the texture graph one would discard edges between an R-tree vertex and any of its descendants because the descendants are spatially superimposed over the parents. Then, region growing is done in exactly the same way as before, but now, similarities among texture elements at different levels of the R-tree can be identified automatically.

(4) Other clustering algorithms

The reason for selecting the MST technique for similarity grouping is that Zahn has noted the ability of such an algorithm to simulate Gestalt grouping. Burr and Chien [Bu76] also used the MST technique as a tool for grouping the elementary square regions in a scene. Their procedure is developed based upon _a priori_ knowledge which is not a general approach to low level processing. Clustering algorithms other than the MST method might also work well provided that they can be extended to multidimensional attribute space. A broad survey can be found in Ball [Ba65]. In particular, Jarvis and Patrick [Ja73] use shared nearest neighbors for a similarity measure; Gowda and Krishna [Go78] use the concept of mutual nearest neighborhoods; Narendra and Goldberg [Na77] developed a homogeneity measure called directed trees for image segmentation. Jarvis [Ja77] also applied the shared nearest neighbors to feature sets extracted from color imagery for image segmentation. A generalized k-nearest neighbor rule is proposed by Patrick and Fisher [Pa70]. An algorithm for finding nearest neighbors can be found in Friedman, *et al* [Fr75].

REFERENCES

[Ba76] Bajcsy, R. and L. Lieberman, "Texture gradient as a depth cue," Computer Graphics and Image Processing (5), March 1976, pp. 52-67.

[Ba65] Ball, G.H., "Data analysis in the social sciences: what about the details?" Proc. Fall Joint Computer Conf., 1965, pp. 533-559.

[Br70] Brice, C. and C. Fennema, "Scene analysis using regions," Artificial Intelligence (1), 1970, pp. 205-226.

[Br66] Brodatz, P., Textures, Dover, New York, 1966.

[Bu76] Burr, D.J. and R.T. Chien, "The minimal spanning tree in visual data segmentation," Proc. Third International Joint Conf. on Pattern Recognition, 1976, pp. 519-523.

[Ca77] Carlton, S.G. and O. Mitchell, "Image segmentation using texture and gray level," Proc. Pattern Recognition and Image Processing Conference, June 1977, pp. 387-391.

[De73] Deutsch, E.S. and N.J. Belknap, "Texture descriptions using neighborhood information," Computer Graphics and Image Processing (1), August 1972, pp.145-168.

[Dy79] Dyer, C.R., T.H. Hong, and A. Rosenfeld, "Texture classification using gray level cooccurrence based on edge maxima," Technical Report TR-738, Computer Science Center, University of Maryland, March 1979.

[Eh76] Ehrich, R.W. and J.P. Foith, "Representation of random waveforms by relational trees," IEEE Transactions on Computers (25), July 1976, pp. 725-736.

[Eh78] Ehrich, R.W. and J.P. Foith, "A view of texture topology and texture description," Computer Graphics and Image Processing (8), October 1978, pp. 174-202.

[Fr75] Friedman, J.H., F. Baskett, and L.J. Shustek, "An algorithm for finding nearest neighbors," IEEE Transactions on Computers (24), October 1975, pp. 1000-1006.

[Go78] Gowda, K.C. and G. Krishna, "Disaggregative clustering using the concept of mutual nearest neighborhood," IEEE Transactions on Systems, Man, and Cybernetics (8), December 1978, pp. 888-895.

[Ha73] Hall, D.J., D.A. Huffman, R.O. Duda, and D.E. Wolf, "Development of new pattern recognition methods," NTIS AD-722 614 November 1973.

[Ha73] Haralick, R.M., K. Shanmugam, and I. Dinstein, "Textural features for image classification," IEEE Transactions on Systems, Man, and Cybernetics (3), November 1973, pp. 610-621.

[Ja73] Jarvis, R.A. and E.A. Patrick, "Clustering using a similarity measure based on shared near neighbors," IEEE Transactions on Computers (22), November 1973, pp. 1025-1034.

[Ja77] Jarvis, R.A., "Computer image segmentation: first partitions using shared neighbor clustering," NTIS PB-277 929, December 1977.

[Ka55] Kaizer, H., "A quantification of textures on aerial photographs," Technical Note 121, Boston University Research Laboratories, 1955, AD69484.

[Kr71] Krakauer, L.J., "Computer analysis of visual properties of curved objects," MAC TR-82, Massachusetts Institute of Technology, May 1971.

[La79] Lai, P.F. and R.W. Ehrich, "Segmentation of images with incompletely specified regions," IEEE Transactions on Systems, Man, and Cybernetics (9), December 1979, pp. 864-868.

[Lu78] Lu, S.Y. and K.S. Fu, "A syntactic approach to texture analysis," Computer Graphics and Image Processing (8), June 1978, pp. 303-330.

[Mu68] Muerle, J.L. and D.C. Allen, "Experimental evaluation of techniques for automatic segmentation of objects in a complex scene," in G.C. Cheng, et.al. (eds.), Pictorial Pattern Recognition, Washington: Thompson, 1968, pp. 3-13.

[Na77] Narendra, P.M. and M. Goldberg, "A graph-theoretic approach to image segmentation," Proc. Conference on Pattern Recognition and Image Processing, 1977, pp. 248-256.

[Pa70] Patrick, E.A. and F.P. Fischer, III, "A generalized k-nearest neighbor rule," Information and Control (16), April 1970, pp. 128-152.

[Pr57] Prim, R.C., "Shortest connection networks and some generalizations," Bell System Technical Journal, November 1957, pp. 1389-1401.

[Ro70] Rosenfeld, A. and E. Troy, "Visual texture analysis," Proc. Symp. on Feature Extraction and Selection, October 1970, pp. 115-124.

[Ro71] Rosenfeld, A. and M. Thurston, "Edge and curve detection for visual scene analysis," IEEE Transactions on Computers (20), May 1971, pp. 562-569.

[Ro77] Rosenfeld, A. "Fuzzy digital topology," Technical Report TR-573, Computer Science Center, University of Maryland, September 1977.

[Sa79] Sankar, P.V. and A. Rosenfeld, "Hierarchical representation of waveforms," IEEE Transactions on Pattern Analysis and Machine Intelligence (1), January 1979, pp. 73-80.

[To73] Tomita, F., M. Yachida, and S. Tsuji, "Detection of homogeneous regions by structural analysis," Proc. Third Int. Joint Conference on AI, 1973, pp. 564-571.

[Wa78] Wang, S., "Structural representation of textures," M.S. Thesis, Computer Science Department, Virginia Tech, July 1978.

[Ya73] Yakimovsky, Y. and J. Feldman, "A semantics-based decision theory region analyzer," Proc. 3rd IJCAI, October 1973, pp. 580-588.

[Za71] Zahn, C.T., "Graph-theoretical methods for detecting and describing gestalt clusters," IEEE Transactions on Computers (20), January 1971, pp. 68-86.

[Zo75] Zobrist, A.L. and W.B. Thompson, "Building a distance function for gestalt grouping," IEEE Transactions on Computers (24), July 1975, pp. 718-728.

[Zu75] Zucker, S.W., A. Rosenfeld, and L.S. Davis, "Picture segmentation by texture discrimination," IEEE Transactions on Computers (24), December 1975, pp. 1228-1233.

[Zu76] Zucker, S.W., "Toward a model of texture," Computer Graphics and Image Processing (5), June 1976, pp. 190-202.