STRUCTURAL PROCESSING OF VISUAL INFORMATION

by

R.W. Ehrich[1]

J.P. Foith[2]

DEPARTMENT OF COMPUTER SCIENCE

VIRGINIA POLYTECHNIC INSTITUTE

AND STATE UNIVERSITY

March, 1977

[1]R.W. Ehrich is on leave from the University of Massachusetts.

[2]J.P. Foith is with the Institute für Informationsverarbeitung, Fraunhofer Gesellschaft, Karlsruhe, Germany.

ABSTRACT

One lesson that has been learned from previous approaches to scene analysis is that local methods are insufficient for extracting reliable information about the contents of a scene. Two different procedures that have been tried in order to remedy this deficiency are the use of knowledge via *a priori* information and internal models and multilevel analysis based on hierarchies of representations such as cone systems. It does not seem appropriate to drive the very first levels of analysis by *a priori* knowledge. It is doubtful that it will be possible to use knowledge in a way general and versatile enough to direct low level processing, and there is a need for some powerful data driven mechanisms that might at a later stage invoke internal models. It would seem more appropriate to obtain some crude global information through glancing or planning at low resolution levels that can drive a more scrutinous analysis at high resolution levels. While hierarchal systems are therefore good, the way they are currently being constructed is not necessarily good.

In this context the issue of low level representation becomes more and more important, and not enough attention has been paid to this issue. Even Marr's provocative ideas about his primal sketch do not go to a sufficient level of analysis, and it is felt that more of the workload should be thrown onto the first processing levels. In this paper is posited a comprehensive hierarchal data structure that requires no decisions and therefore no parameters for its construction. The technique does not require preselected windows, but rather uses context-dependent criteria. The data structure is versatile, easily computed, and invertible in the sense that the original image is completely recoverable.

INTRODUCTION

A major problem that has been perplexing computer vision theorists is that there appear to be so many useful operators for extracting scene information from the intensity array and so many ways to represent and combine this information. Indeed, the multitude of different approaches suggests that we may still be looking for the wrong things. An analysis of our problems should begin with the issues of low level representation, which deal with the transition from the raw intensity array to the basic internal representation within the machine. Hanson and Riseman[1] have concluded that a number of intermediate data structures are necessary for storing the information required for semantic processing. For preliminary processing they use a cone system that provides scene information at varying resolution levels. Then they propose an RSE data structure [2] that contains hypotheses abou regions and boundaries that have been obtained from the cone system by a collage of operators. Marr[3] has proposed dispensing with the intensity array entirely by forming a primal sketch that contains a record of preliminary statistical interpretations of directional intensity events in the intensity array.

Such data structures have many desirable properties. For example, the upper layers of cone systems provide interpretations of global events, and both the lower layers of cone systems and the primal sketch provide interpretations of intensity changes similar to those that are known to occur at early stages in the visual system. One particular difficulty with cone systems, however, is that the information represented in the higher levels is difficult to use directly. One reality that one has to live with is that the results will never be better than the represenatation that is used. In the case of the human visual system, the number of retinal detec-

tors outnumbers the number of fibers in the optic nerve by at least two orders of magnitude, though the number of detectors in the foveal region is comparable to the number of fibers. Whatever the encoding of visual information, the original scene is completely recoverable later in the visual system. The visual data are stored multiply in different layers in the brain, and there are hints that at least some of these representations preserve the topology of the retinal stimuli. Furthermore, whatever the retina tells the visual cortex about intensity changes in the scene, the luminance information is also transmitted. At this point very few representations have been proposed, and the problem of selecting representations has not even been thoughtfully discussed in the literature. Typically intensity arrays are transformed into gradient images, numerical approximations, cooccurrence matrices, frequency images, cones, or symbolic arrays. Unfortunately, the underlying issues about the use of such representations have rarely been discussed. Instead, much effort has been put into the evaluation of the results of the techniques that make use of these representations. In any case, the issue of which scene representations should be used in computational vision theory is very much open.

In our work we have long been dissatisfied with the performance of edge and bar shaped filter masks in edge and line detection applications. Curiously, such filters with discontinuous receptive fields have Fourier transforms that contain many zeros in the finite frequency plane. Consequently, if one wished to reconstruct a scene that has been filtered by such a mask, one would find that the inverse reconstruction filter has many singularities. Of course, one could alter the filter masks so that no information is lost, but the idea of requiring the human visual system to perform deconvolution is unacceptable. Since the human visual system does

not appear to have frequencies of zero transmission, the representations it uses are probably more complex than those that are in common use.

There seem to be a number of reasons why such filter masks have found considerable following. For one, such detectors seem to model the detectors that are believed to exist in the neural layers behind the retina. For another, such detectors are reminiscent of some powerful theorems about matched filters that at first glance would appear to be applicable. Third, edge and bar detectors are a delight to implement, even though their outputs might not be easy to interpret.

Regarding the first point, it is clear that the visual system must contain differencing circuitry with which to detect intensity changes. However, little is known about the actual neural implementations, and equally little is known about how luminance information is encoded in the transmissions from the retina. The second point requires somewhat more elaboration. The theorems about matched filters are derived about known signals in isolation under conditions of disturbances with known statistics. In Figure 1a is shown a scan line across an image that contains white lines normal to the scan direction. One possible method for detecting these lines is to apply to the scan line bar filters such as those shown in Figures 1c and 1e. The positive outputs of these filters are given in Figures 1b and 1c for the two different filter parameters; these outputs represent two different interpretations of Figure 1a. The problem with the outputs of these filters is that they are context dependent in a very complicated way; the close proximity of two lines substantially affects the filter outputs for both. Since the smaller filter makes use of less context, its output is a more reliable representation of the scan line structure. In any case, such filters must be used with great care, and their use probably should not be regarded as an application of matched filtering. A more detailed discussion

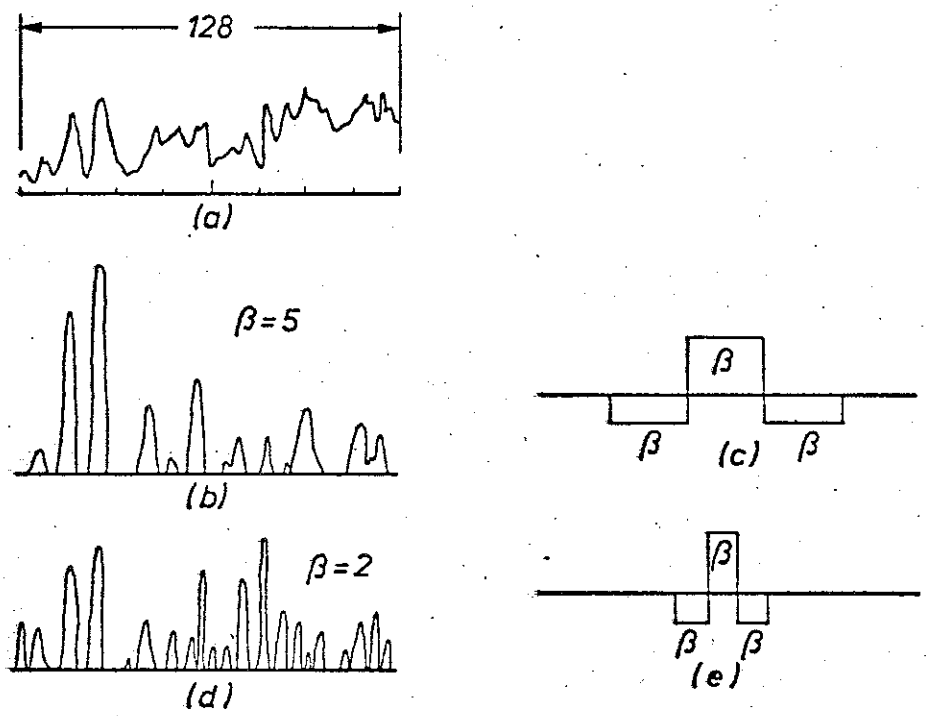


Figure 1 - Scan line (a), filter output (b) for filter mask (c), and filter output (d) for filter mask (e).

of this issue can be found in [4].

The remaining point for discussion is the one about the ease with which bar and edge type filters can be implemented. It is very easy to become dogmatic about "nice" techniques with fast running speeds that almost work, but in a field as difficult as computer vision, one must be prepared to appreciate the successes of "near misses," enumerate the shortcomings, and try something else. It is the purpose of this paper to propose a technique that produces a highly informative scene representation; it is a structural technique in the sense that it analyses the topology of the picture function without generating interpretations. It finds and represents the structures that are actually there as opposed to computing a statistical measure of how closely the picture function resembles the shapes of some preselected masks. This is a technique that we believe can achieve many of the results that have already been achieved while providing a framework for doing with relative ease some things that were difficult before. Moreover, this paper

concerns a philosophy about making decisions in hierarchal systems that Marr has called the "Principle of Least Committment," hereafter denoted the PLC [3].

Those of us who have designed a complex system with hierarchal decision making are aware of the way in which an insignificant decision at the bottom of the hierarchy can topple the entire system if made irrevocably but incorrectly. The PLC states that one should never make a decision based upon thin evidence that will later have to be changed because of the consequences ot that decision farther up the hierarchy. In other words, to the maximum extent possible, decision making elements should be placed in a feedback loop so that firm decisions are deferred until the evidence becomes compelling. This is an excellent framework for a relaxation process - one keeps all options open until the system stabilizes in an optimal configuration.

The construction of a hierarchal system becomes much more difficult when highly interpretive processes are active even at the lowest processing levels because there is the possibility that a firm but incorrect decision might contradict the correct response of the system. The PLC provides for the inclusion of the correct interpretation among the alternatives produced at each decision stage. To minimize the number of alternatives that must be explicitly retained, it is essential to transform the information at the earliest stages in such a way that the difficulty of making decisions is minimized while at the same time no viable alternatives are removed from consideration. For this reason we consider the early representation of scene information to be a most critical issue in the conceptual design of an automatic scene analysis system.

Marr forms his primal sketch by applying edge and bar filters to a raw image and including in the sketch a summary of the significant information in the outputs of these filters. While Marr's illumination of the issues of low

level vision represents an important contribution to the field, we question the wisdom of using from the outset a representation that appears to violate his own PLC. We believe that by using different techniques it is possible to form a much more accurate and sophisticated representation of the original scene deterministically so that the PLC is not violated at the representation level.

STRUCTURAL PROCESSING OF DIGITAL PICTURE FUNCTIONS

In 1971 at MIT, Krakauer [5] wrote a program to compute contour maps for digital picture functions. Conceptually, all one need do is move an imaginary plane downward through the graph of the picture function and for each intensity, plot the closed curves formed by the intersection of the plane with the graph. Krakauer noticed that as the plane was lowered, the regions thus delineated would be contained within one another in a way that could be conveniently represented by a tree. Aside from the computation of a few region shape descriptors, nothing of much interest resulted from that work.

For several years Pavlidis [6] has been interested in approximating functions such as picture functions by segments of polynomial surfaces using a technique known as split and merge. In split and merge one typically begins with a tentative segmentation; for each region one computes the best polynomial surface and then the corresponding figure of merit. Regions with poor matches are split into smaller regions, and adjacent regions that are fit by surfaces with similar parameters are merged until a satisfactory segmentation has been determined. The technique fails, however, to represent any relationships among the regions of the segmentation and is primarily a means of establishing a simpler description of the surface topology. The principal drawback of such a technique in scene analysis is that the coefficient labels attached to the surface segments may not have much meaning in the image domain, although the original image will be recoverable to any desired accuracy. Therefore, invertibility is not the only requirement for a good scene representation. The primitives of the representation must have a clearly defined relationship to the original image and to each other, and the relationships must themselves convey useful information. In this paper a question is also raised about the sufficiency of simple adjacency relationships.

Other investigators have sensed that significant information can be obtained directly from the topology of the picture function, though no particular theories have evolved. For example, Enomoto and Katayama [7] have done formal work using differential geometry to establish some rigorous definitions for topological features such as ridges and edges. As is frequently the case, inherently one-dimensional techniques may be used to obtain quasi-two-dimensional results. Lozano-Perez [8], noting that a substantial amount of information about a scene was contained in individual scan lines or intensity profiles, developed a method for detecting objects by using a syntactic description of intensity profiles across the object. The syntax was stored in the form of an ATN (augmented transition network) grammar for which a scan line parsing algorithm was constructed. Such a technique does not lead to any useful representations for general scenes since it is an object-specific technique, but it reinforces other investigators' suspicions that the picture function topology can be used directly.

Since 1975, Ehrich and Foith [9,10] have been investigating picture processing algorithms using a scene description based on a data structure for representing intensity profiles that is called a relational tree or simply, R-tree. An R-tree is a tree data structure that is a complete description of the one-dimensional contextual relationships defined by the peaks and valleys of an intensity profile. Pointers from the tree vertices link the structural elements to attribute lisis from which the original scene can be completely reconstructed. Since this data structure is the key concept for this entire paper, a brief review of relational trees is given next.

In Figure 2 is shown a simple intensity profile containing a few peaks and valleys. Notice that peaks appear recursively nested within larger and larger peaks that might be concatenated with and nested within even larger peaks. The peaks themselves are delineated by the highest of the valleys on

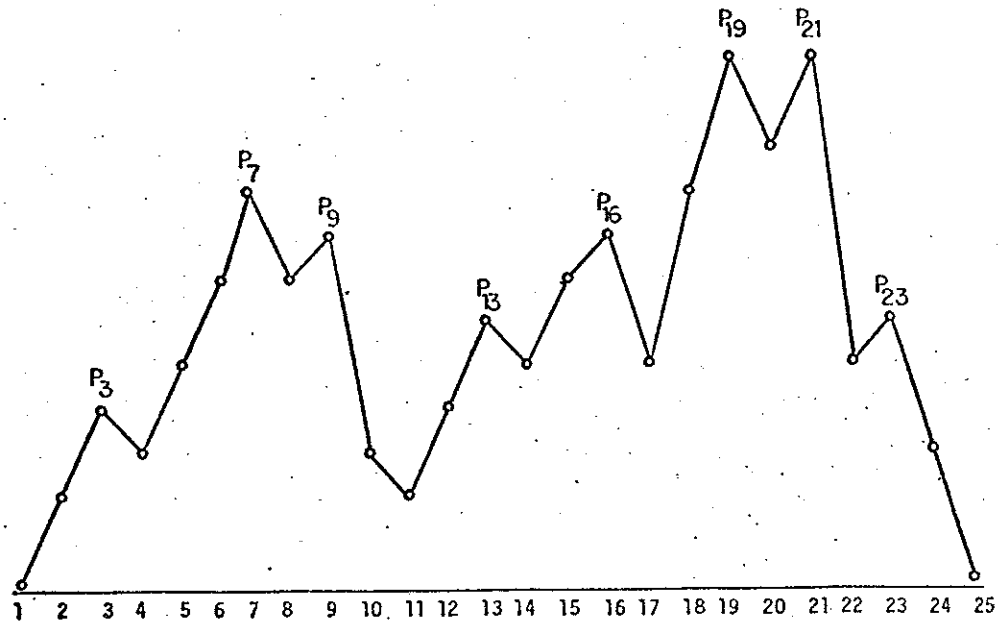


Figure 2 - Sample random waveform.

either slope. For example, the small peak p_7 is concatenated with the small peak p_9, and both together form a larger peak that is delineated by the valley at data point 4 and whose width extends from point 4 to point 10. Since p_7 is the higher (or dominant) peak, this larger peak is conveniently labeled p_7. Now, this recursive nesting of waveform peaks is conveniently reflected in a tree, called a relational tree, that is given for this example in Figure 3.

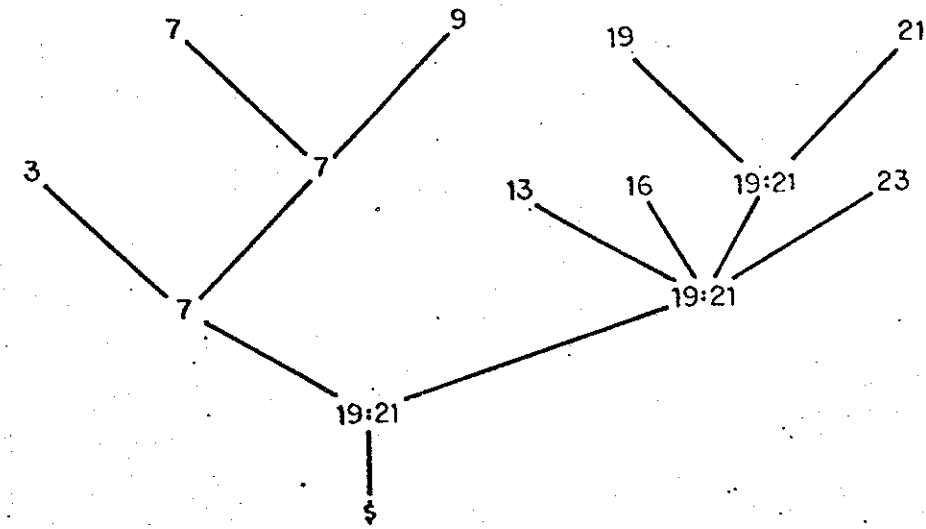


Figure 3 - Relational tree for Figure 2.

In this tree, several vertices are labeled 19:21 due to the fact that the

dominant peaks in part of the waveform are p_{19} and p_{21} since both have the same height. Ther vertex with four descendants is due to the fact that the valleys at data points 14, 71, and 22 are all at the same height. Each non-frontier vertex of the R-tree is physically due to a unique valley of the intensity profile, and it is labeled with that peak among those induced by the valley that is dominant. The vertices in the frontier of the tree is a left to right ordering of all the local intensity maxima of the profile.

The R-tree is computed using a fast 3-stack algorithm that processes an intensity profile from left to right. During processing the algorithm produces a great deal of information about the peaks and valleys of the profile, and this is stored in attribute lists that are attached to the vertices of the R-tree. In our implementation the widths, heights, and locations of peaks are stored together with valley locations and pointers that permit quick traversal of the tree.

It is easiest to regard the representation of an image as a list of R-trees as though the representation had been produced by a transformation, τ_t, called the tree transformation. Our first concern is with the invertibility of τ_t, since if the original image cannot be recovered, τ_t is interpretive rather than representational. In Figure 4a is an image of a pile of capacitors, and in Figure 4b is the reconstruction of the image from the data structure. Notice the legibility of the code on the capacitor at left center and the crisp object boundaries. To give some feeling for the algorithm itself, Figure 5 shows scan line 136 in Figures 4a and 4b in addition to the R-tree for that line.

While the original concept was developed mainly for intensity modulated textures, it was clear that for the representation of homogeneous regions an extension of the concept would be necessary. The reconstruction in Figure 4b was generated by drawing straight line segments between adjacent peaks

and valleys. The reason that the reconstruction is so accurate is that the digitization noise and small intensity variations in the regions that appear to be uniform cause the profile to be reconstructed by many more shorter line segments than would otherwise be used. To illustrate this,

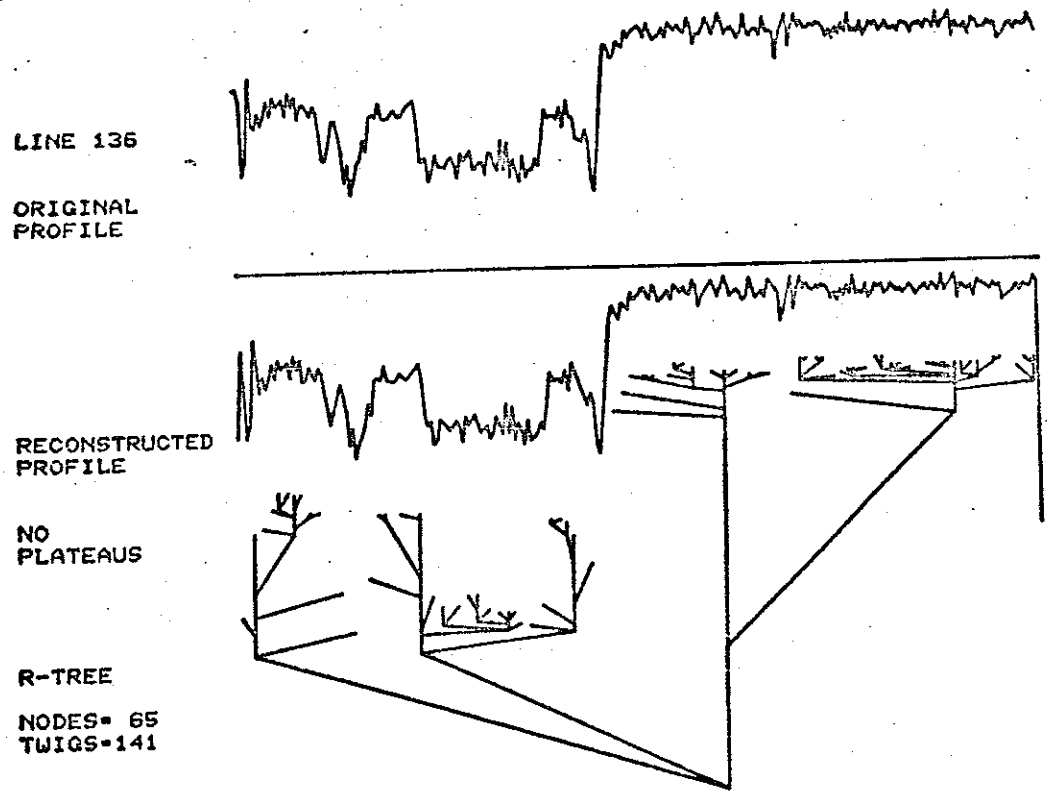


Figure 5 - Scan line 136 of Figure 4 and R-tree.

in Figure 6a the scan lines of the image in Figure 4a have been smoothed by a special hysteresis algorithm [11] that has eliminated small peaks and valleys while preserving exactly the shape and locations of all the major intensity maxima and minima. In this case the hysteresis was set at 40 intensity units out of a possible 256 so that peaks and valleys of amplitude less than 40 would be removed. The striking similarity of this image with the original in Figure 4a despite the amount of processing that this image has undergone suggests that most of the small amplitude image detail is irrelevant for the purposes of scene analysis. Figure 6b shows the reconstruction from the data structure, and Figure 7 shows scan line 136 in

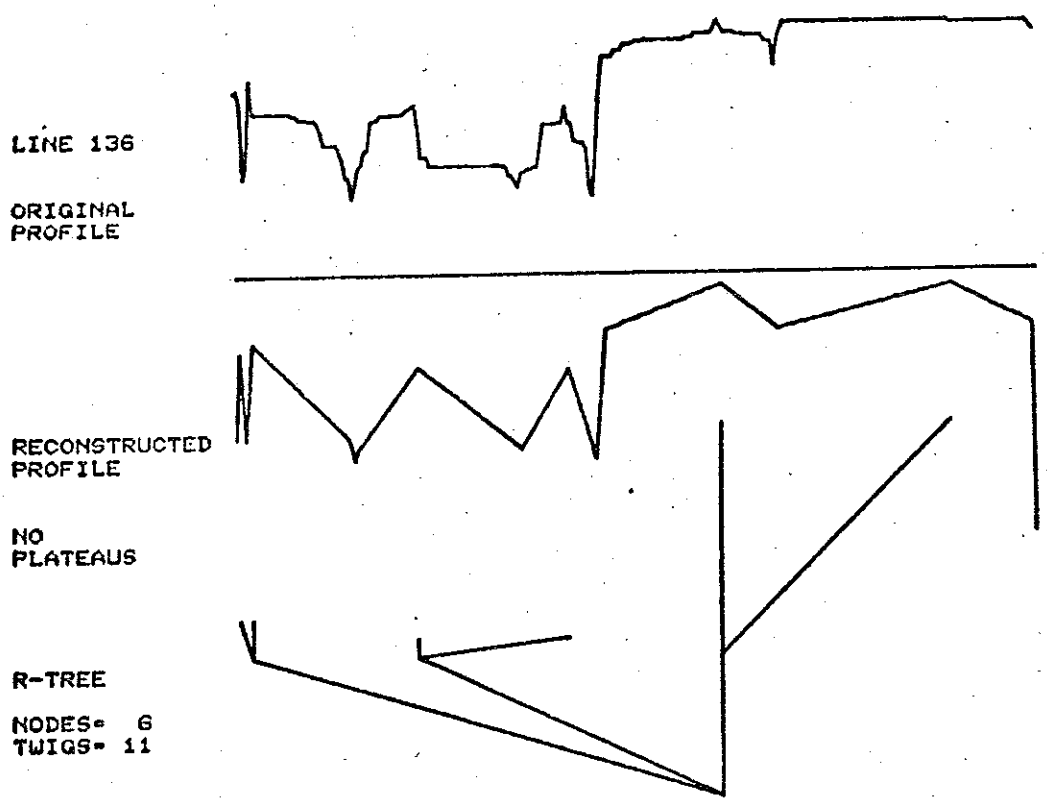


Figure 7 - Scan line 136 of Figure 6 and R-tree.

Figures 6a and 6b in addition to the R-tree for that line. Since all small intensity variations have been removed there is insufficient information in the data structure to accurately determine the boundaried of the larger regions.

The problem of representing homogeneous regions was solved by detecting regions of zero slope, called plateaus, and storing them in the data structure. If one considers the plateaus to be peaks of zero relative height, the basic algorithm is not changed. In fact, one might also consider the possibility of quantizing slope and storing in the data structure regions of constant slope. The reconstruction of Figure 6a using the tree algorithm with plateaus is shown in Figure 8. Notice the dramatic improvement in the representation. Once again, Figure 9 shows scan line 136 in Figures 6a and 8 in addition to the R-tree for that line. No effort has been made to aggregate adjacent plateaus at different heights, since such simple

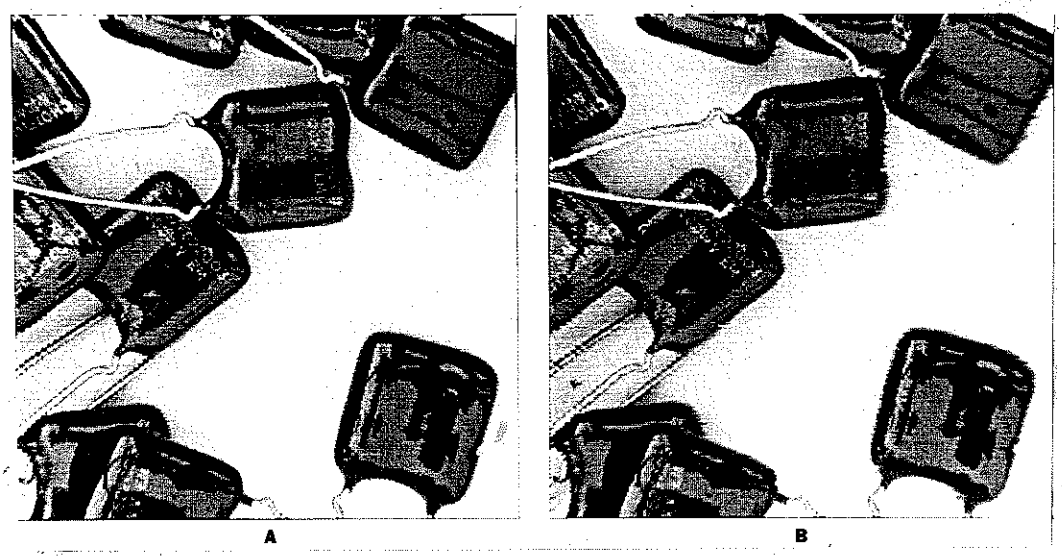


Figure 4 - Raw image (a) and reconstruction from the data structure (b).

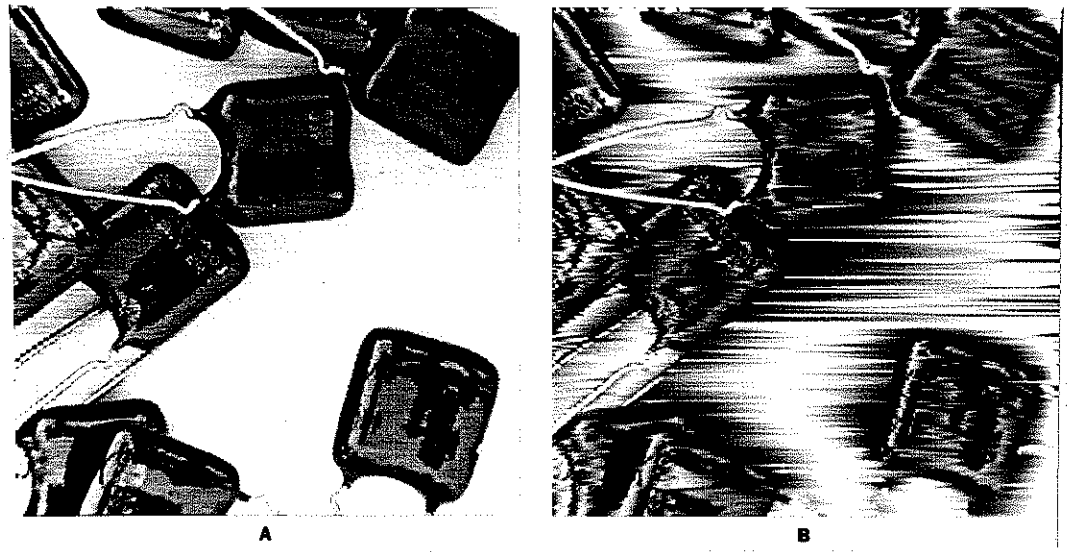


Figure 6 - Image after hysteresis smoothing of 40 units (a) and reconstruction from the data structure (b).

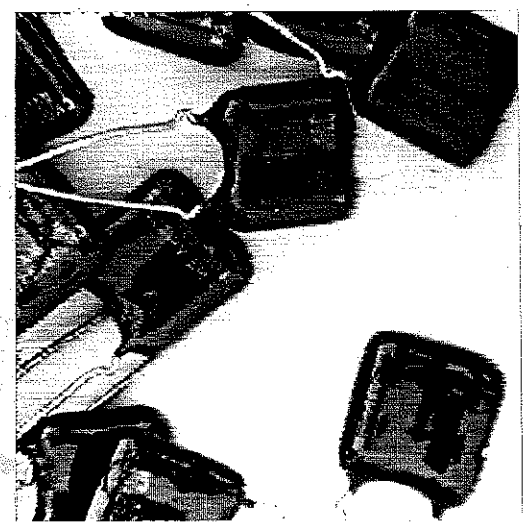


Figure 8 - Reconstruction of Figure 6a using plateaus.

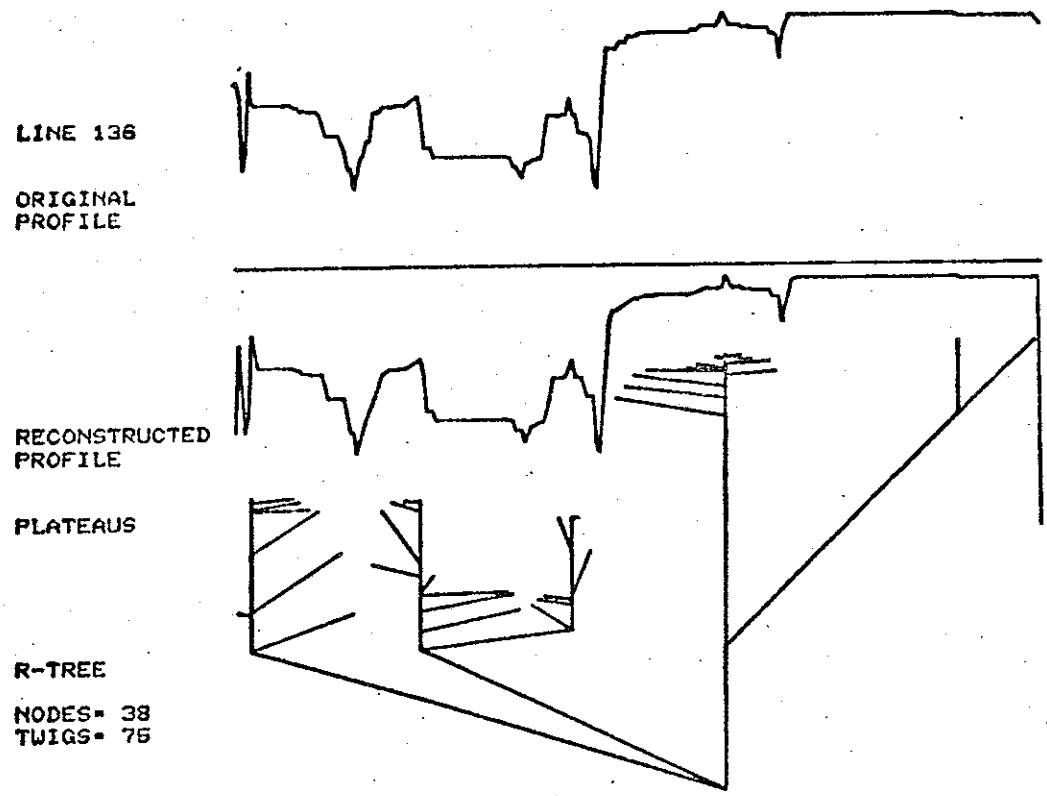


Figure 9 - Scan line 136 of Figures 6a and 8 and R-tree.

interpretive processes should occur farther along in the processing sequence. Instead, every effort has been made to ensure that there are absolutely no parameters at this stage of visual processing.

One of the useful properties of R-trees is that large intensity peaks that correspond to large one-dimensional regions are located near the root of the R-tree while their substructures are located closer to the frontier. Therefore one can choose to ignore image substructures that are judged too insignificant to be of importance in the problem at hand. On the other hand, the presence of these substructures in the data structure permits very precise description of large image structures without the loss of resolution that is characteristic of systems employing windows.

Since grouping is the real task of a vision system, it is still necessary to show the types of processing that can be accomplished with this repre-

sentation and to show that useful two-dimensional aggregation can be achieved. In the last section a complete representation based on R-trees is posited, and it is shown that one of the principal advantages of an approach that is inherently one-dimensional is the way in which one can describe directional information in the picture function.

GROUPING

One of the most difficult and important problems is to determine how picture elements should be aggregated into regions. Aggregation processes may be entirely data driven or entirely directed by prior knowledge; typically they are directed by both. Although we hardly know anything about how the visual system uses prior knowledge, it is a reasonable guess that the lowest level processing algorithms are not modified by prior expectations. Whether this is the case or not, attempts to construct a hierarchal vision system would be greatly complicated if no knowledge-independent low level processing was permitted. In our view, the basic scene representation is used precisely to organize the results of knowledge-independent processing. Later a number of more complex grouping processes are activated simultaneously, taking as input the basic representation. Their outputs compete and cooperate to produce regions at higher and higher levels of structural complexity.

It is almost certainly possible to achieve better knowledge-independent results than we have achieved in the past, and it is important to do so to provide the semantic processing routines with the best possible input. The primitives of the basic scene representation should organize picture elements of the intensity array into gestalts such as segments of constant slope, curvature, intensity, or whatever attributes are considered to be important. Marr, of course, has made a strong case for directional microedges and slope segment boundaries; these, however, are linear primitives whose only relationships are concatenation, and they do not account for luminance information in any useful way. We have selected the next most feasible primitive called a peak because the gestalt it represents is a self-contained intensity region in the picture function. Peaks have the advantage of representing luminance information directly, and since they are self-embedding, their relationships are much richer and more descriptive. Once a primitive has been selected,

the remaining issue involves the way in which it is computed. Here again we differ with Marr because, as was stated earlier, we believe that the actual instances of the primitives should be recorded rather than correlations with a set of ideal primitives.

Almost all previous work on edge detection is based upon the use of edge models, which in many applications is a most reasonable approach. However, one obtains different representations by using different models; since in a general system one cannot make assumptions about the nature of the data, the need for preselecting a set of models is distressing. A better approach would be to provide a representation in which any particular model can be tested directly, and we believe that our representation is especially suited for that. From our point of view, edges are derived structures rather than primitives and are a consequence of the ordering of picture element intensities on the sides of peaks and valleys. Moreover, a great deal of contextual information is available that would, for example, make it easy to determine that a given edge bounds a telephone pole rather than a house.

To demonstrate the viability of the representation, let us consider one of the simplest possible structural edge detection algorithms. In Figure 10a is shown a well known house scene that contains both sharp boundaries and heavy texture. For comparison, the output of the Sobel operator is given in Figure 10b thresholded at intensity 40 which is close to the level at which the roof-sky boundary begins to disintegrate. Figure 10c shows the result of applying a simple structural edge detector independently to row and column profiles of the image and then merging the results into a new pseudo-image. The edge detector functions in the following way. From the R-tree of each profile the slope segments between each pair of adjacent plateaus, peaks, and valleys are located. Suppose that $p_1=(x_1,y_1)$ and $p_2=(x_2,y_2)$ are the endpoints of such a slope segment on a given intensity

Figure 10 - House scene (a), Sobel operator with threshold 40 (b), structural edge detector with threshold 40 (c), and structural edge detector with 3 pixel blurring normal to the profiles and with threshold 40 (d).

profile. A file is generated that contains the value $|y_2-y_1|$ at an arbitrarily selected coordinate $x=(x_1+x_2)/2$, and in Figure 10c are shown those edge points that exceeded a threshold of 40.

One might argue against the advisability of making decisions about the presence of edge points using context only in the profile direction because one is ignoring the usual semantic requirement of edge continuity.

There are several possibilities here. One would be to check adjacent profiles before making the edge assertion. Another way of introducing context normal to the profile has some of the spirit of the Sobel operator itself. One blurs vertically before detecting horizontal edges and horizontally before detecting vertical edges so that when a slope segment is located in the R-tree for a given profile it already implies that the slope segment is continued in the adjacent profiles. Since the blurring occurs before the computation of the R-trees, this process is external to the basic representation that has been discussed up to this point; however, it is fundamental to the total representation to be discussed in the next section because the R-trees of directionally blurred images contain direct representations of directional image structures. The result of edge detection after directional blurring is shown in Figure 10d for blurring with a sliding average of 3 picture elements out of a total scan line length of 256 elements. Once again, a threshold of 40 has been selected.

This structural edge detector produces edges that do not appear quite so straight as those of the Sobel operator because they are so thin and because the edge elements are always located half way between extrema. On the other hand, a greater responsibility has been given to the system designer to specify what he means by an edge -- in fact, all options are left open. For example, in this implementation, a long slope that is interrupted by a short plateau will be interpreted as two smaller edges. Sufficient information is present in the R-trees to facilitate much more complex interpretations, and adjacency of multiple edges does not confuse the detector in the least. Notice how the Sobel operator glues the window boundaries and porch supports into ambiguous aggregates, while the structural detector produces rather clear separation of the significant slopes. Notice also that as anticipated, the contextual algorithm produces fewer noise points

than the line by line algorithm.

In another experiment we have analyzed the magnitudes of the slopes of the slope segments detected by the structural edge detector and have not found the information to be of any value. It may be that slope information will be helpful for identifying shadings produced by curved objects, but apparently the edges occurring in a scene like this are all quite sharp. In such images it is purely edge contrast that is critical to interpretation.

In the work that we have done to date we have concentrated much more on region growing than on edge detection because the peak gestalt embodies simultaneously the notions of region and edge boundary. In fact, if it were possible to achieve proper aggregation of peaks into regions, edge boundaries could be determined easily after aggregation rather than before. The first step is to consider the one-dimensional region structure within an intensity profile, since R-trees contain a complete record of the contextual relationships of peaks and valleys within individual profiles. As was previously noted, an R-tree segments an intensity profile into recursively nested regions according to the relationships among the intensity maxima and minima within the profile. These are tentative region components whose identity will depend in large part upon the presence or absence of corresponding structures in adjacent propiles. In [9] we investigated the possibility that there were fundamental structural transformations that map R-trees into R-trees with simpler structure by grouping peak substructures in a useful way. Two of particular significance are the splinter and trunk transformations, so named after the particular subtrees required to generate transformed R-trees directly from the original R-trees. Omitting the theoretical discussion of these transformations, the effect these transformations have on the surface structures of intensity profiles can be rather easily described.

Algorithm 1 - Splinter Rule

Locate in the intensity profile all maximal sets of consecutive peaks whose valleys strictly ascend in height from left to right and then strictly descend. The peaks in each such set are merged into a new peak whose height and location is that of the dominant (highest) peak in that set.

Algorithm 2 - Trunk Rule

As in the splinter rule, locate all maximal sets of consecutive peaks whose valleys strictly ascend in height from left to right and then strictly descend. For each such set, mark the peak, p_0, between the ascending and descending valleys. For each such set, move one by one from the highest valley to the lowest, marking the peaks for each new valley until either all peaks in the set are marked or dominance changes from p_0 to another peak. For each set, merge the marked peaks into a new peak whose height and location is that of p_0.

The effects of the splinter and trunk rules are demonstrated in Figure 11 for line 9 of the smoothed image in Figure 6a. Notice that the trunk rule results in less aggressive aggregation because the groups of peaks to be merged are split by dominance changes. Notice also that these grouping rules act on peaks, leaving valleys unchanged. Thus these rules act to aggregate white figures on a black background, and if one wished to aggregate black figures on a white background, one would first form the negative of the image. Trivial modifications can be made to the splinter and trunk rules so that after grouping the resulting profiles more closely resemble the shape of the original profiles. These modified rules, called the modified splinter and trunk rules, are shown in Figures 11e and 11f. The modification requires that in the reconstruction of the profile the new profile follows the tops of the peaks that are being merged; since additional valleys are being introduced, the modified profiles have different trees

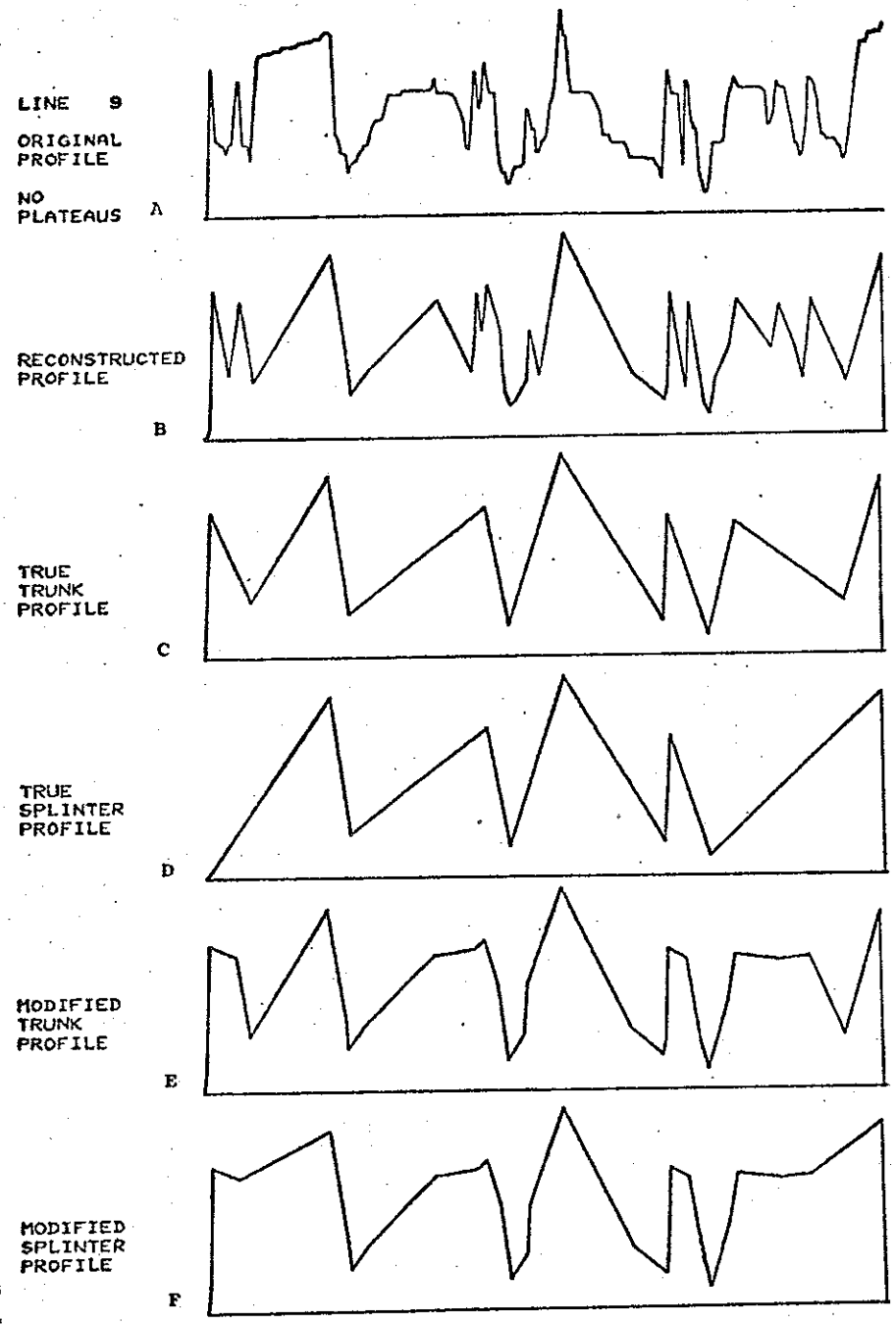


Figure 11 - Scan line 9 of Figure 6a (a), reconstruction (b), trunk rule (c), splinter rule (d), modified trunk rule (e), and modified splinter rule (f).

than do the profiles produced by the original splinter and trunk rules.

The remarkable thing about these grouping transformations is that they can achieve limited but useful two-dimensional results even when applied to intensity profiles independently. In actual use one would constrain the grouping by imposing semantic constraints. The results of applying

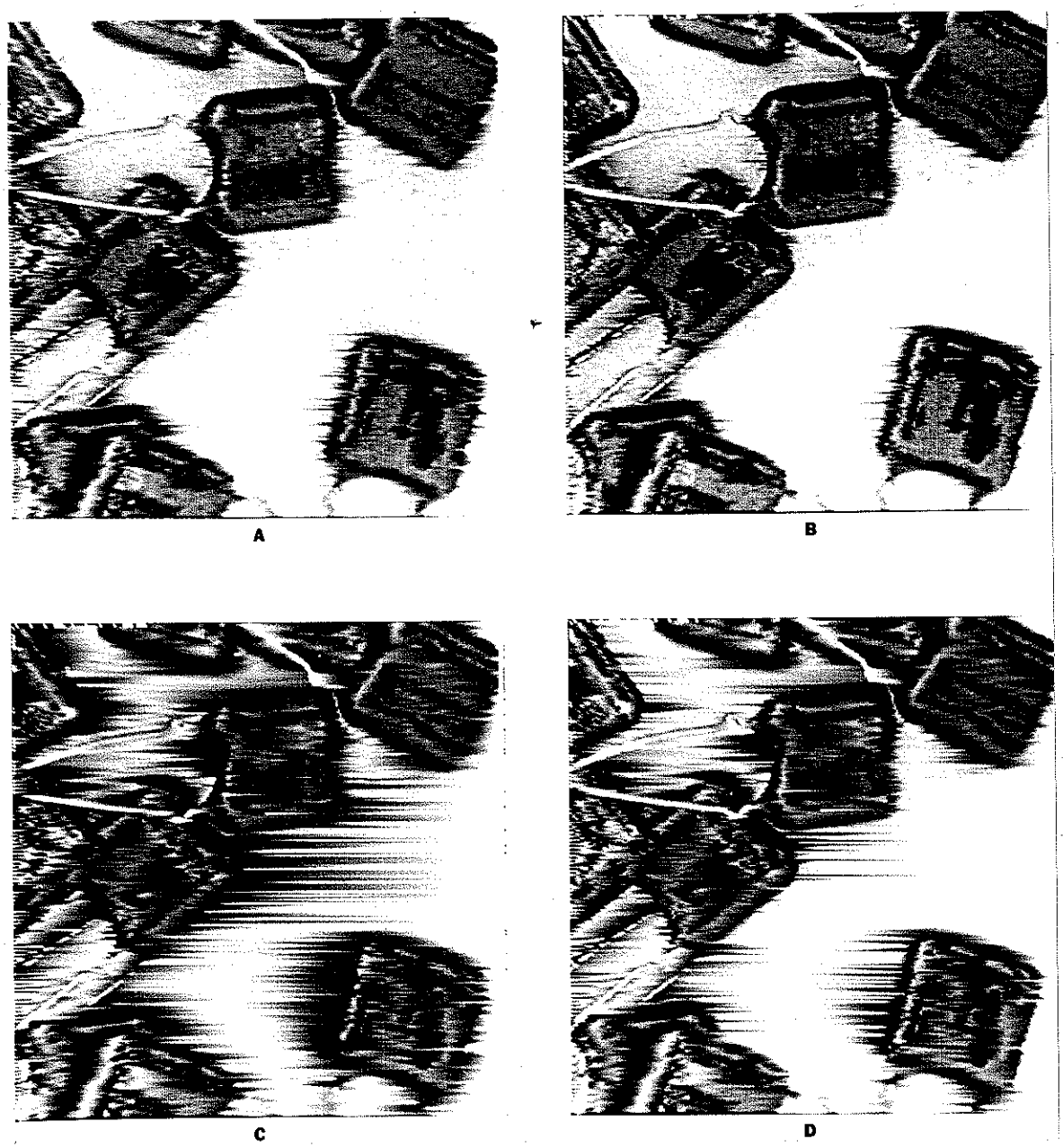


Figure 12 - Splinter rule (a) and trunk rule (b) applied to inverted capacitor scene, splinter rule (c) and trunk rule (d) applied to smoothed, inverted capacitor scene.

the splinter and trunk rules to the scan lines of the inverted capacitor image in Figure 4a are shown in Figures 12a and 12b. Aggregation is relatively slight (notice the blurring of the characters on the capacitors) because the small peaks and noise points in the raw image are removed first. If the splinter and trunk rules are applied to the smoothed image of Figure 6a, the results in Figures 12c and 12d are obtained. The severe blurring is due to the procedure of reconstructing a profile by connecting peaks

and valleys with straight line segments. This is the motivation for using the modified splinter and trunk rules, which construct more acceptable profiles.

In Figure 13 is shown a collage of scenes obtained by iteratively applying the modified splinter rule of the inverted, smoothed capacitor scene of Figure 6a. Notice in the capacitor at the lower right how the white holes due to reflections gradually fill in. Since in the inverted image the black capacitors are figure, the black regions gradually take over the white regions, filling in first the smaller and then the larger holes. For comparison, consider a similar collage in Figure 14 generated by the modified splinter rule on the uninverted scene. In this case the white background is considered by the algorithm to be figure, and gradually the capacitor in the lower right is being absorbed by the white area. To ensure that the grouping rules act on the perceptible regions, both experiments were run by starting with the smoothed image.

While these simple aggregation mechanisms are inadequate in themselves for region growing, they demonstrate that simple hypotheses about regions can be formed according to criterea that are entirely structural in nature. As mentioned, planning will be helpful for determining how many peaks should be grouped at each merge step, and as the regions grow more complex, semantics will surely have to be applied to the process. Most important, grouping must also make use of context in the direction normal to the intensity profiles, and later on some methods are explored for accomplishing this. For the moment, however, let us explore some of the other properties of peaks and valleys.

In an earlier investigation of a set of texture samples [9] it was discovered that the histograms of one-dimensional extrema (peaks and valleys of row and column profiles) tended to be nearly identical to the histograms for

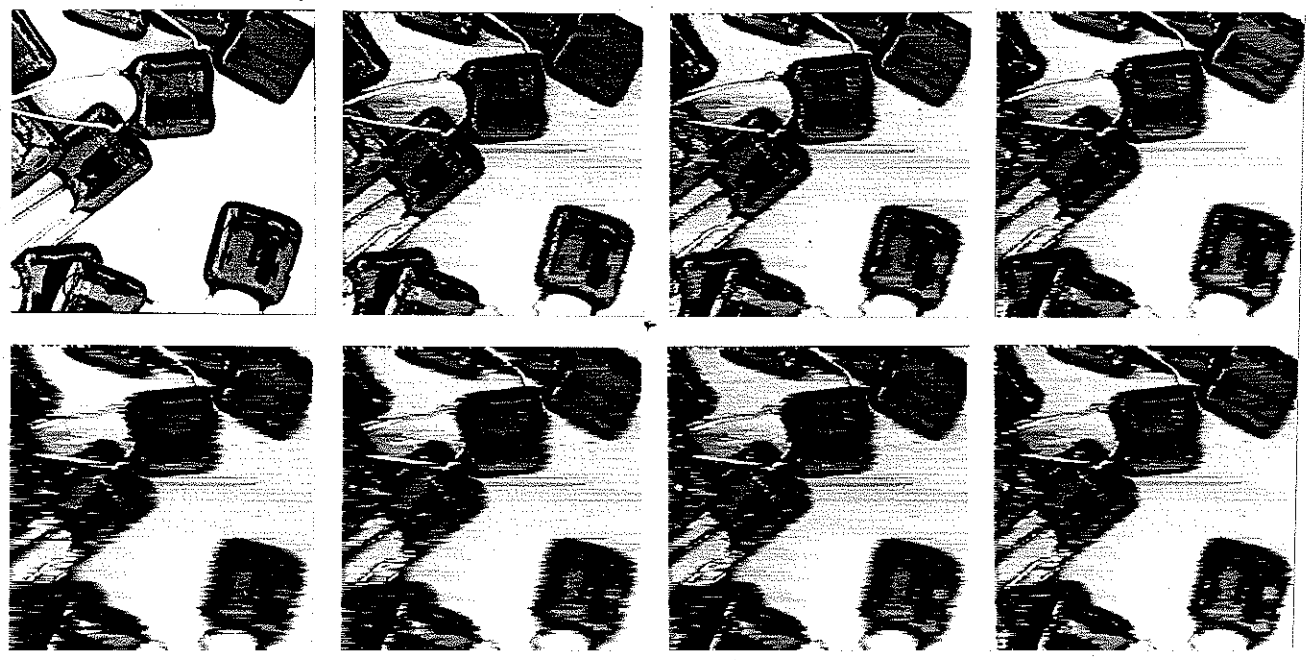


Figure 13 - Iteratively applied modified splinter rule (clockwise). Black regions are treated as figure.

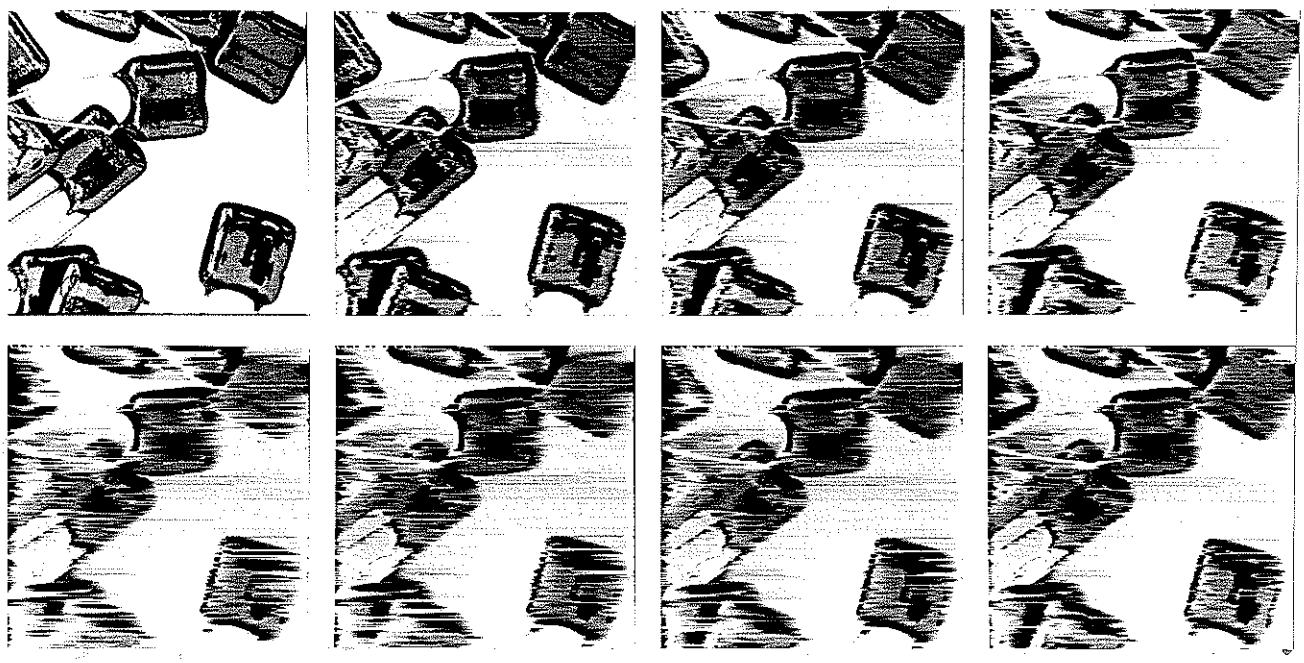


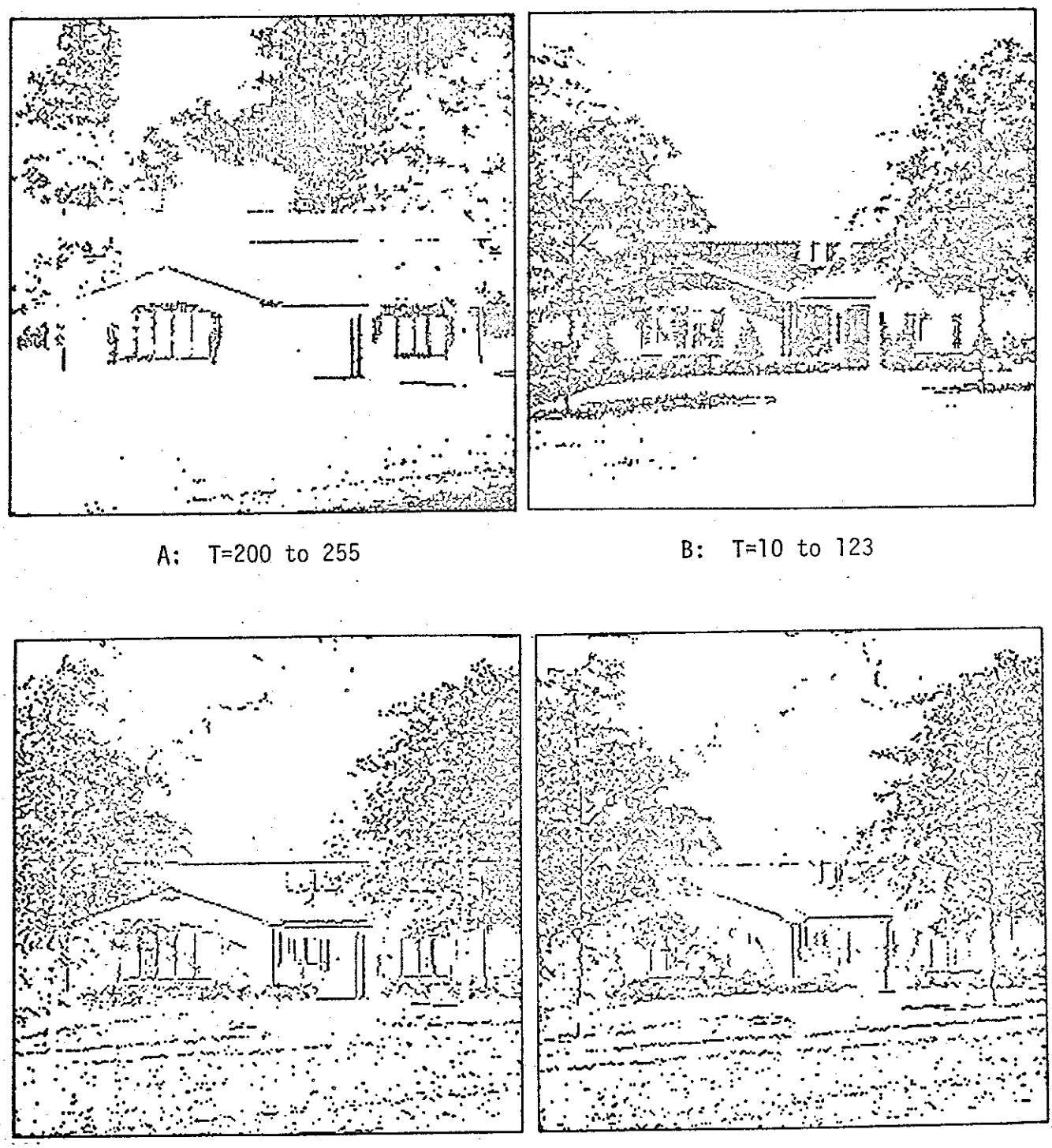
Figure 14 - Iteratively applied modified splinter rule (clockwise). White regions are treated as figure.

all picture elements, almost as though the picture elements had been inserted for cosmetic effect. However, the histograms for peaks alone or for valleys alone tended to differ considerably both from each other and from the histo-

gram for all picture elements. It occurred to us that it might be possible to segment regions of heavy texture on the basis of peak and valley attributes, and while we had evidence that this was indeed the case, we never looked at the results that might be achieved when working with a complex scene. In Figure 15 are slices from four pseudo-images generated from the house scene in Figure 10a. Figures 15a and 15b contain all one-dimensional peaks and all one-dimensional valleys, respectively. To obtain a simple measure of texture contrast we plotted in Figure 15c the relative heights of peaks on the frontier of the R-trees. Relative peak height is the vertical distance between the top of a peak and the valley that induces it, and this information is part of the R-tree data structure. Figure 15d is a plot of the relative peak heights (ie., valley depths) in the negative image.

There is a vast amount of information in Figure 15 about textured regions. Most of the important regions form an isolated point cluster in one or another of the four slices. Linear features such as tree trunks, branches, and house boundaries are strongly visible and supplement edge detection results. One should compare carefully the cloud regions in Figure 10a with those in Figure 15a. It appears that many textures may be characterized by point clusters in three-dimensional (x,y,intensity) space where the peaks and valleys fall. What is really needed are some good algorithms for detecting colinearities and for bounding these point clusters. All the necessary feature measurements are readily accessible in the R-trees.

Let us now turn briefly to one final topic. The theory that has been discussed is fundamentally a two-dimensional theory, whereas the implementation is strictly one-dimensional. While two-dimensional information is not lost in the data structure, it is not there explicitly; one reason for not using a data structure that describes two-dimensional regions directly is the potential complexity of such a structure. For example, it would be necessary



A: T=200 to 255

B: T=10 to 123

C: T= 25 to 255

D: T=25 to 255

Figure 15 - 1D peaks (a), 1D valleys (b), 1D relative peak heights (c), and 1D relative valley depths (d).

to describe saddle points as well as regions that contain holes. It is possible, however, to link the vertices of the R-trees of adjacent scan

lines to generate two-dimensional regions.

In Figure 16, four sequential scan lines have been plotted from an inverted, low resolution (64 x 64 element) copy of the capacitor scene in Figure 4a. Lines 17 to 20 in that scene fall across the capacitor in the

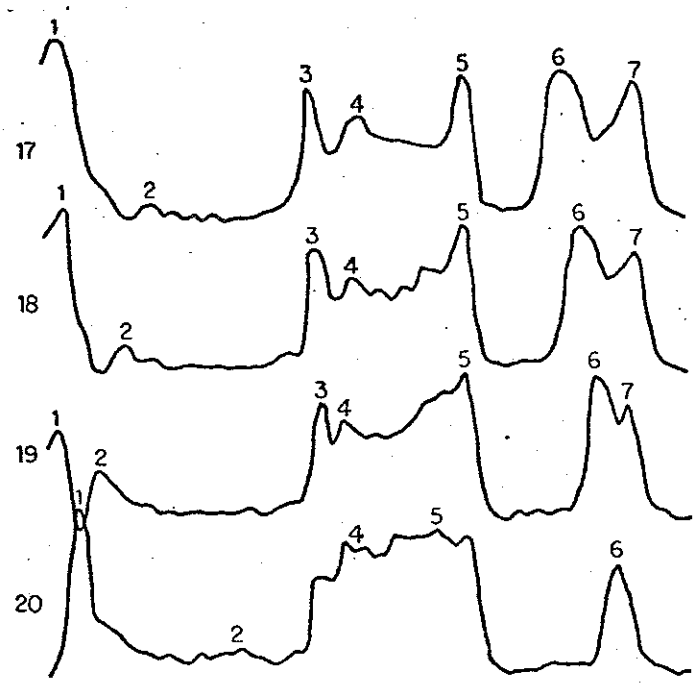


Figure 16 - Scan lines from Figure 6a.

upper center of the photo, and the three major peaks correspond to capacitor bodies. The R-trees for scan lines 17-20 have been plotted in Figure 17. In each of these four scan lines, the largest region that covers the entire scene is represented at the root of each corresponding tree. Moving up each tree one now finds vertices that correspond to smaller regions labeled 1, 5, and 6. These correspond with the outlines of the three capacitors crossed by the scan lines. Moving still higher in the trees one finds the vertices of still smaller subregions that correspond with specular reflections from the capacitor surfaces. The process of region growing in the direction normal to the scan lines involves linking vertices of corresponding structures.

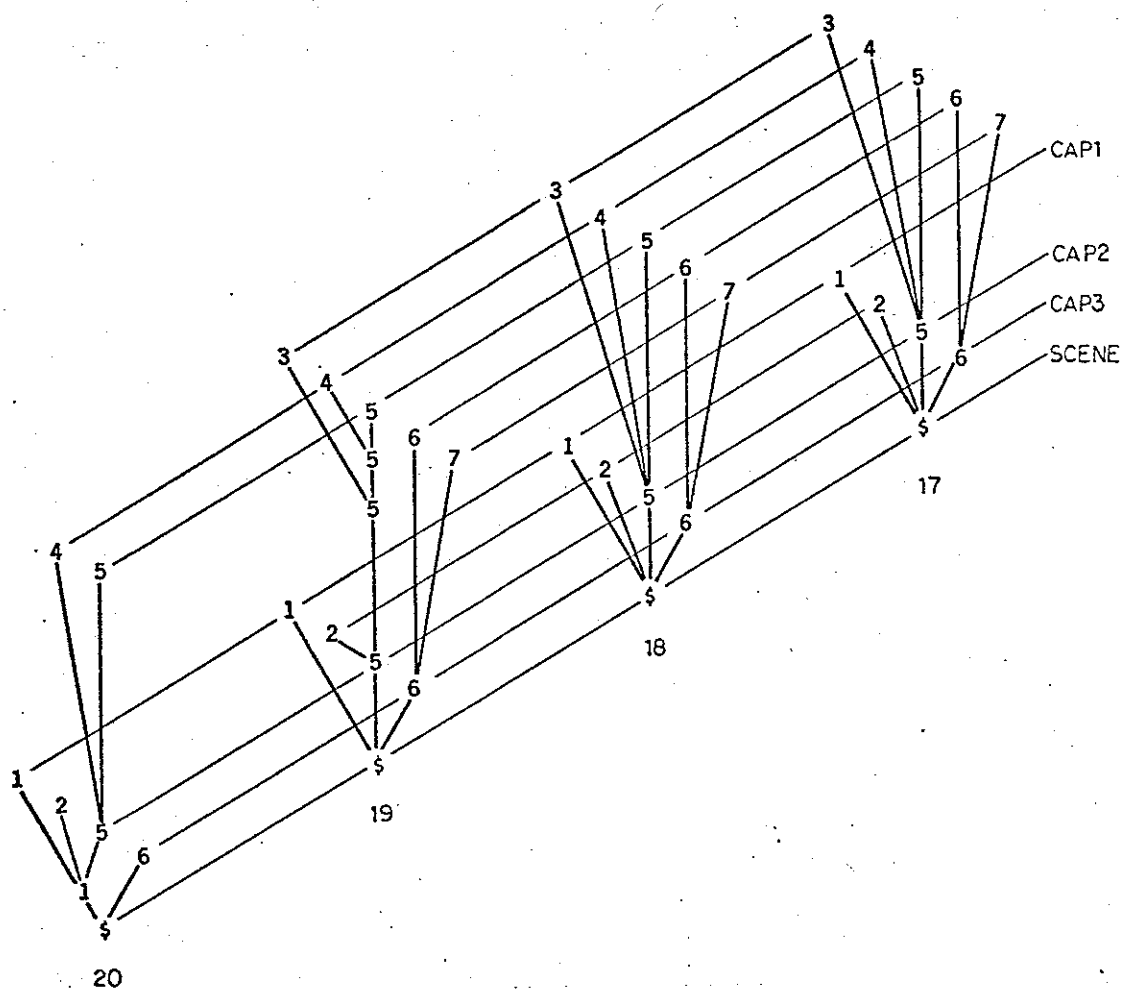


Figure 17 - Coupled relational trees for scan lines 17-20.

Clearly not all vertices in all trees will link to adjacent scan lines, and it is necessary to decide not only how a vertex links but whether or not it links at all. The procedure is to match scan line surface structures by a discrete relaxation process as shown in the example in Figure 18. Establishing a link between peaks establishes a link between the frontiers of the corresponding R-trees. On the other hand, establishing a link between valleys induces a link between vertices in the deep structure of the R-trees. Fortunately, the information in the R-tree data structure provides all the necessary feature measurements on peaks and valleys on which the relaxation can be based.

The linked list of R-trees is an approximation to a data structure that describes the exact two-dimensional topology of the picture function.

For example, the regions are recursively nested in the scan line direction, whereas they are only concatenated in the normal direction. Nevertheless, true two-dimensional region growing seems eminently practical, and structural grouping mechanisms can be devised to operate on the linked regions, rather than on the independent scan lines.

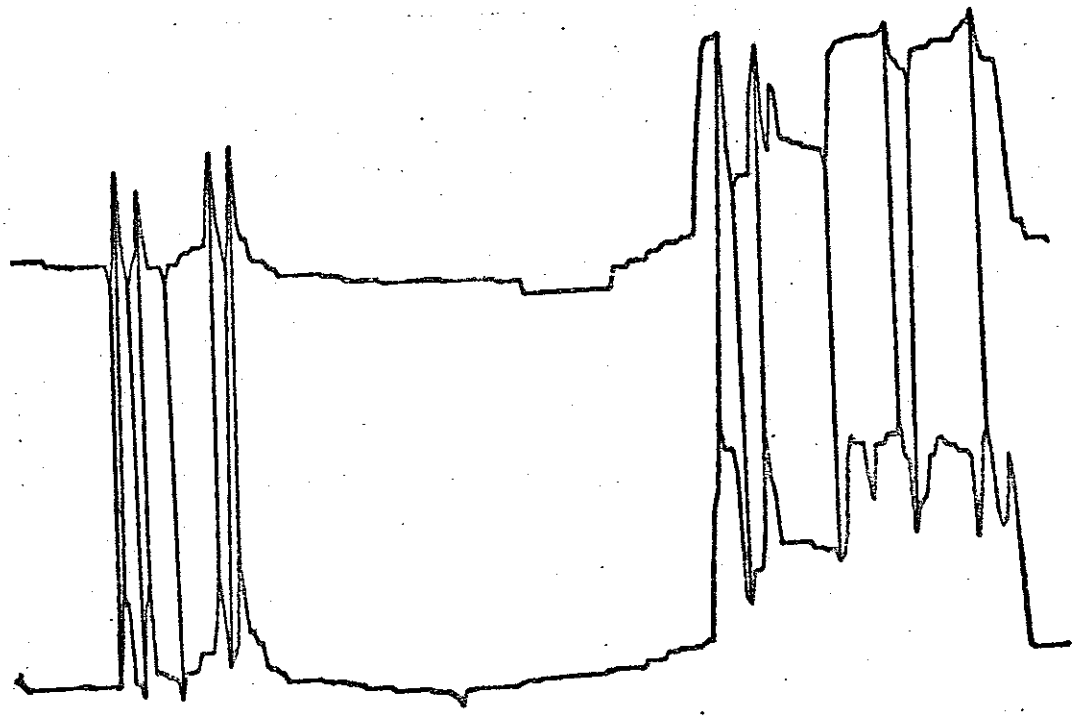


Figure 18 - Linked surface structures.

SCENE REPRESENTATIONS

The purpose of a viable scene representation is to provide a data structure that describes in a useful way the intensity changes in a scene and their interrelationships. There are at least five criterea by which such a representation should be selected.

1) What should be the primitives of the representation? One must decide what kinds of information will be essential to the analysis tasks later on and what supplementary information will facilitate those tasks.
2) By which techniques should the primitives be extracted? There appear to be two major classes of techniques here -- those that are statistical (interpretational) in nature and those that are structural (representational).
3) What kind of data structure is required? The data structure determines the ease with which scene information can be retrieved and expresses the essential interrelationships among the primitives.
4) What are the aggregation modules for which the scene representation is to serve as input? If, for example, the formation of a line drawing of the region boundaries is to be fundamental to the analysis, then the representation ought to facilitate the formation of the boundaries. If, on the other hand, region growing is to be the primary goal, one ought to consider representing the information that is essential for deciding region composition.
5) How are the aggregation modules going to function? This has been one of the most difficult problems associated with low level processing. We simple don't know the best way to form boundaries or the best way to bound regions of heavy texture, and many investigators believe that each aggregation module will really consist of a set of submodules that produces a rich collage of alternative interpretations. On the other hand, the types of submodules that can be implemented, or at least the

difficulty with which they can be implemented, depends a great deal upon the representation.

Aside from the arguments in favor of our particular choices and implementation, it is felt that at this level the data structure should be representational and not interpretational. It should have a form that is useful for the processing steps that follow, and, in order to make scene analysis less of an art and more of a science, it should not be necessary to use a different representation for each different application, as seems to be the case at the present time. While computational expediency may dictate the use of a simpler representation for tracking bubble chamber tracks than for analyzing a LANDSAT frame, the theory, at least, should be the same.

We would like to include here a few remarks on our choice of peaks as the primitives for the representation. Since a peak is basically a two-dimensional region, the use of such a representation leads to a region-oriented as opposed to a boundary-oriented theory. However, as we have demonstrated, boundaries are part of the concept of a peak, and they can be determined explicitly from the data structure. When one looks at a gradient image the brain deceives us into believing that we have found the object boundaries by aggregating edge assertions for us. The representationand, for that matter, the edge image in Figure 10c are merely reports about what is in the data, and the really hard job, which is what we understand least, is to use this information for aggregation. As to the importance of local intensity maxima and minima, it is unknown whether they play roles of corresponding importance in the human visual system.

There are a few other considerations that have not yet been discussed. For example, while directional intensity changes can be tracked in the trees, it is not at all clear that this would be the best approach. The reason that R-trees do not record directional information is simply that they are

computed from one-dimensional data. Additional information can be obtained by blurring the image directionally first and then computing its R-trees. Since in a blurred image the R-trees are not independent, not all of them need to be stored. Directional blurring is identical to the concept of directional averaging functions[12] that were used for detecting very subtle lines and edges in remoto reconnaissance, and this is the same process that was used to obtain the edge detection results in Figure 10d.

Another issue that has not been resolved concerns figure-ground decisions. Peaks and valleys are dual structures, although they have not been given identical treatment. 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 R-trees of the original and directionally blurred images, but also the corresponding ones for the inverted image. Of course, all the information is implicit in the original list of R-trees, but one should not neglect the amount of computation required to retrieve it.

The most general form of the representation we propose is therefore a set of R-trees, not only for the row and column profiles and their complements, but also for the image after it has been selectively blurred in various directions. The number of blurring directions and the degree of blurring required is determined by how difficult the analysis tasks are, but the concept underlying the representation is application independent. The R-trees for the blurred images will provide a very sharp and precise description of long lines or edges whose gradients are normal to the blurring direction. The number of orientations required for the blurring process is related to the degree of blurring, and that relationship is defined more precisely in [12].

After the computation of the tree representations, an interpretive phase is initiated whose result will be a number of competing and cooperating results that will be stored in a layered data structure. Here the sorting of information and the resolution of conflicts begins. The interpretations stored here are the result of analysis of the trees for edges, lines, atomic regions, and so on, tracking for extension to two dimensions, aggregation by mechanisms such as splinters and trunks, and aggregation of heavy textures. Evidence has been shown that operations like edge-detection, region growing, texture analysis, and even glancing (by looking at deeper tree vertices) can be accomplished by using the R-trees. However, there is much work to be done in developing effective mechanisms for these specific tasks, and it is felt that the selection of a proper representation will be a solid advance toward these goals.

REFERENCES

1. Hanson, A.R. and Riseman, E.M. Processing cones: a parallel computational structure for scene analysis. COINS Technical Report 74C-7, University of Massachusetts, Amherst, 1974.

2. Hanson, A.R. and Riseman, E.M. Representation and control in the construction of visual models. COINS Technical Report 76-9, University of Massachusetts, Amherst, July 1976.

3. Marr, D. Early processing of visual information. AI Memo 340, MIT AI Laboratory, Cambridge, December 1975.

4. Ehrich, R.W. Detection of global linear features in remote sensing data. Proc. Joint Workshop on Pat. Rec. and AI, Hyannis, Massachusetts, June 1976, pp. 51-57.

5. Krakauer, L.J. Computer analysis of visual properties of curved objects. MIT Technical Report TR-82, Project MAC, May 1971.

6. Pavlidis, T. and Horowitz, S. Segmentation of plane curves. IEEE Transactions on Computers, vol. C-23, August 1974, pp. 860-870.

7. Enomoto, H. and Katayama, T. Structural lines of images. Proc. 2nd USA-Japan Computer Conference, 1975, pp. 470-474.

8. Lozano-Perez, T. Parsing intensity profiles. AI Memo 329, MIT AI Laboratory, Cambridge, May 1975

9. Ehrich, R.W. and Foith, J.P. A view of texture topology and texture description. ECE Department Technical Report, University of Massachusetts, July 1975.

10. Ehrich, R.W. and Foith, J.P. Representation of random waveforms by relational trees. IEEE Transactions on Computers, vol. C-25, July 1976, pp. 725-736.

11. Ehrich, R.W. A symmetric hysteresis smoothing algorithm with lookahead. Computer Science Technical Report, Virginia Polytechnic Institute and State University, Blacksburg, May 1977.

12. Ehrich, R.W. Detection of global edges in textured images. IEEE Transactions on Computers, vol. C-25, in press.