COINS TR 78-2

Segmentation of Natural Scenes[1]

Allen R. Hanson
School of Language and Communication
Hampshire College
Amherst, Massachusetts 01002

Edward M. Riseman
Computer and Information Science
University of Massachusetts
Amherst, Massachusetts 01003

Abstract

The extraction of information from a two-dimensional image which is sufficient for constructing a description of the image is a complex task. The VISIONS system employs two distinct parallel iterative segmentation algorithms. The first utilizes local views of the image to aggregate edges into boundaries, while the second utilizes global histograms and a local spatial analysis to form regions with much of the textural detail suppressed. After merging these results into a single representation, features are extracted to produce a description of each region and boundary segment, including two-dimensional shape attributes. This information is symbolically labelled and passed to interpretation processes.

Table of Contents

I. Introduction

The structure of a developing system, known as VISIONS (Visual Integration by Semantic Interpretation Of Natural Scenes), for understanding relatively unconstrained images of natural scenes is outlined in this paper and in a companion paper elsewhere in this volume. The focus of this paper is on segmentation processes, often referred to as "low-level" vision, which partition large amounts of visual sensory data (derived from a static color image) into organized syntactic units which form the basis for further processing. The second paper describes the interpretation processes, often referred to as "high-level" vision, which receive the segmented data and attempt to construct an interpretation in the form of a description of the physical world portrayed in the scene.

The general goal of the low-level system is the transformation of a large spatial array of pixels (i.e., picture elements) into a more compact description of the image in terms of visually distinct syntactic units and their characteristics, including location. By a variety of means, the visual information in the image must be aggregated, labelled with symbolic names and attributes, and then interfaced to higher level knowledge structures. The syntactic units considered here are boundary segments (connected sets of edges between pixels) and regions (connected sets of pixels) whose attributes include relevant properties of color, texture, size, shape, location, etc.

[1]This research has been supported by the Office of Naval Research under Grant N00014-75-C-0459.

Copyright © 1978 by Academic Press, Inc.
All rights of reproduction in any form reserved.
ISBN 0-12-323550-2

The low-level system of VISIONS is outlined in Figure 1. The two processes for extracting regions and boundaries both employ parallel, iterative "relaxation" procedures. The boundary formation process produces initial edges from local spatial views of the data, followed by interactions in local contexts which aggregate these edges into more global segments. The region analysis initially forms a set of possible labels for each pixel based upon clusters in global histograms (feature space), and then uses local spatial relationships to aggregate these pixels into regions with minor textural variations suppressed.

After merging the results from these two segmentation processes into a single representation, features are extracted to produce a description of each region and boundary. These features include those obtained by fitting two-dimensional shapes to regions and segments of boundaries, as well as other visual attributes.

To a certain extent it may seem artificial to select any particular dividing line in the sequence of transformations leading to an interpretation of an image. However, the task of developing a system for general visual perception is extremely complex and there is a compelling need to decompose the overall problem into more manageable portions, particularly during exploration and development [HAN76]. The grossest natural decomposition of such a system seems to lie between the processing of two-dimensional image information -- the low-level segmentation processes -- and the inference of three-dimensional hypotheses about the physical world -- the high-level interpretation processes.

This decomposition is also consistent with our belief that segmentation processes should, in many instances, be effective without recourse to semantics. The partitioning of an image (even a nonsense scene) can be based on differences in visual patterns of color, brightness, and texture. It should be noted, however, that there have been successful efforts in effectively integrating semantics into segmentation processes [YAK73, TEN76], and that there are divergent views on the necessity and desirability of the dichotomy between low- and high-level processes [ZUC75,RIS77,FIS78]. Whatever the limitations imposed by the structure of our system decomposition, they should be ameliorated by the inclusion of feedback paths to lock the interpretation and segmentation processes into a closed loop. These paths may be used to provide semantic guidance in the refinement of the image segmentation and to direct further extraction of features. Thus, segmentation which is initially semantic free is followed by a phase of semantically directed segmentation, and then the systems would be intertwined in further cooperative analysis.

The complexity of the data which is to be examined by the segmentation processes has had a significant effect upon the design of those processes. With relatively complex, unconstrained images, any approach to segmentation will be quite prone to error. Highly textured objects (such as trees), shadows and highlights on regular and irregular surfaces, varied and uncontrolled lighting conditions, etc., all contribute to the difficulty of analysis. Few objects/surfaces can be expected to exhibit uniform visual features, and methods for dealing with this variability must be incorporated.

These problems are exemplified by the sequence of photographs of a tree (Figure 2) taken at varying distances. These images also make clear the problem of separating figure and ground as resolution is varied; the point at which the texture elements of the bark should be extracted as regions in themselves instead of contributing to the descriptors of the tree trunk as a whole, is ambiguous. Goal orientation and focus of attention will be quite important in the formation of useful image segmentations. This leaves ambiguous the level of detail which should be pursued in the first stages of non-semantic segmentation. We do not have definitive answers

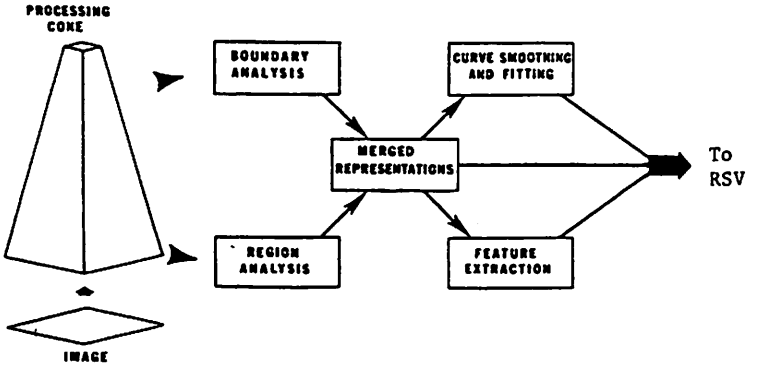


Figure 1. The Segmentation System of VISIONS. Segmentation algorithms and feature extraction processes are applied in a simulated hierarchical parallel array processor. Two distinct parallel iterative algorithms are employed, one to aggregate edges into boundaries, the other to aggregate pixels into regions. The resultant segmentations of the image are merged, regions and boundary segments are smoothed and fitted with simple primitive shapes where possible, and additional features are extracted and associated with each element. The result of this processing is a layered directed graph of regions, segments, and vertices (RSV) which is passed to interpretation processes. These processes involve the construction of a description of the physical world depicted in the image.

Figure 2. Images of a tree at different distances. The resolution of the objects in the image influences the expected segmentation. From our point of view the initial segmentation processes should not be semantically directed, but rather should be based solely upon the relative visual characteristics of the image.

concerning these issues, but our approaches to segmentation have incorporated the representations and flexibility required to move across varying levels of description. The procedure for aggregating edges is structured to easily increase or decrease its sensitivity in the formation of boundaries; in this way the boundary analysis may be set to extract boundaries of any given degree of contrast and/or confidence. The region analysis incorporates a different, but complementary, form of flexibility by producing a hierarchical segmentation down to a varying level of detail. This strategy provides a structure for results which vary according to the needs of the overall system.

II. A Parallel Computational Structure for Processing Images

II.1 Parallel Computation

One characteristic of image analysis which is difficult to ignore is the massive amount of visual data which must be processed. For a full-color image of reasonable spatial (512^2) and color resolution (3 colors, 6 bits/color), close to 5 million bits of information must be processed, often repeatedly. Faced with this data overload, we made a commitment to parallel processing at the very beginning of our research effort [HAN74, RIS74]. If such large amounts of sensory data are eventually to be processed by a machine in close to real time, then the use of large parallel array computers appears to be necessary. It is relevant to note that developments in technology imply that such devices could be economically feasible in the near future.

Given a choice of developing either serial or parallel algorithms (or both), we have developed parallel algorithms wherever possible. A commitment to the discipline of developing algorithms as local parallel operations pays off in providing a way of thinking about transformations of visual data. Many algorithms can be implemented in both sequential and parallel versions. However, in much the same way that language appears to affect thought, thinking in terms of parallel computation leads to the development of algorithms which are often not at all obvious in sequential terms. In addition successful demonstration of segmentation algorithms for simple parallel hardware makes future implementation on real machines far more clear.

Finally, there is a distinct need to reduce the large amounts of visual information, while at the same time extracting features from local areas of the image. Many interesting features (including textural properties) are not a function of individual pixels, but rather a function across the set of points in a local "window" of the image, where the size of this window will vary. Extracting such features would be facilitated by a hierarchical organization of layers of decreasing image (and processing) resolution.

II.2 The Processing Cone: A Hierarchical Parallel Array Computer

We have chosen to simulate a general parallel computational structure for analyzing large arrays of visual data. It is a parallel array computer, called the "processing cone", which is hierarchically organized into layers of decreasing spatial resolution in which information extracted from increasing sizes of receptive fields can be stored and processed further. This structure, which is described in detail in [HAN74] and applied in [HAN75,NAG77], is also related to the recognition cones of Uhr [UHR72], the hierarchical data structures of Klinger [KLI76], the pyramids of Tanimoto and Pavlidis, and Levine [TAN75,LEV78], the computational structure of algorithms developed by Rosenfeld et. al [ROS71,HAY74], the planning algorithms of Kelly [KEL71] and Price [PRI77], and the knowledge-directed analysis of Ballard, Brown, and Feldman [BAL78]. A survey of some of these uses appears in this volume [TAN78].

The function of our processing cone is the transformation and reduction of the massive amount of image data, while at the same time providing a structure in which information at higher levels can direct more detailed processing at lower levels of the cone. The processing cone can be thought of as a simulation of hierarchically organized parallel arrays of microcomputers (Figure 3), where each microcomputer, at a given level, has access to a window of data in a set of planes at the level below it. Note that each level could have an

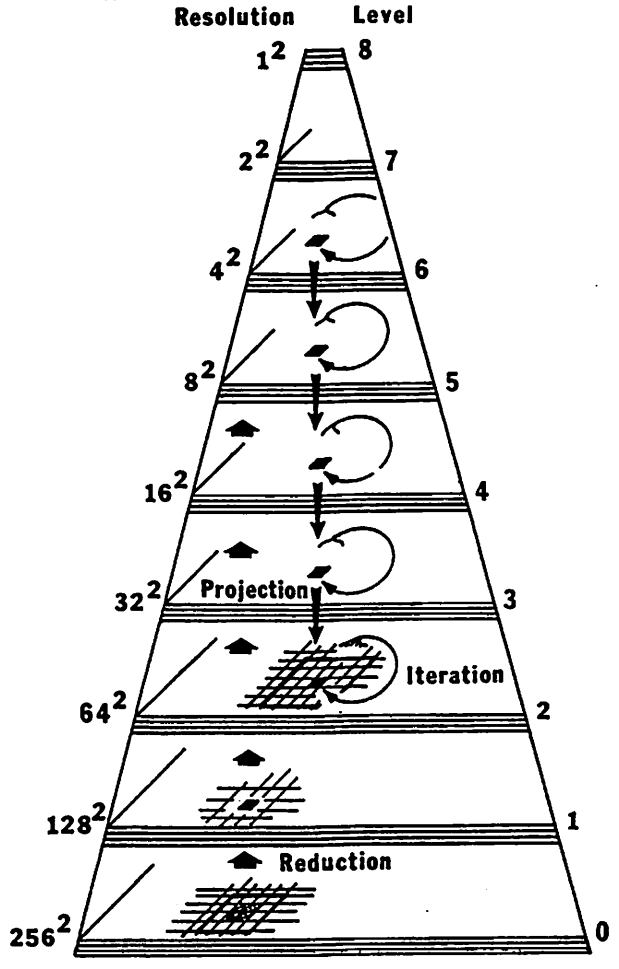


Figure 3. The Processing Cone. This structure is a parallel array of microcomputers hierarchically organized into layers of decreasing spatial resolution. The function of the cone is the transformation and reduction of the massive amounts of image data. Three modes of processing are defined within the cone: reduction of information upward, iteration where information is transformed at a single level, and projection of information at upper layers downward. The actual functions of the data computed during these modes are programmable; each is defined for a local window of data and at some moment in time applied in parallel across all the local windows at a specified level.

arbitrary number of planes in which to store input data, partial results, and final results.

The function computed by each microcomputer is applied in parallel to local windows across the entire array at the specified level. The result obtained from each local window is stored either at the same level or at the next higher level. There are three major modes of operations in the processing cone. During a reduction process upward through the layers in the cone, the data is reduced because portions of each window are nonoverlapping. An iteration process allows the data to be analyzed and/or transformed at a single level of the cone; the size of the array remains constant due to overlapping of windows. A projection process allows information in upper layers to influence computation in lower layers. Thus, information flows up, down, and laterally within the cone. These three modes may be intermixed, providing a rich logical structure in which to implement parallel segmentation algorithms. We will assume that at any moment of time, computation will be taking place at only one level, and therefore the algorithms are defined by a sequence of operations over time, each at a particular level.

While the processing cone has provided a framework in which to think in parallel hierarchical terms, the primary computation of the two segmentation algorithms described in the following sections takes place at only a single level of resolution. There exist a number of possibilities for utilizing the hierarchical structure of the cone. For example, if hierarchical relaxation procedures can be defined [DAV78], then computation at many levels might take place simultaneously. It is also possible to develop planning algorithms which use upper levels of the cone to efficiently direct finer resolution algorithms lower in the cone [KEL71, HAN74,NAG77,PRI77]. It may also be feasible to develop methods for hierarchically refining segmentations in parallel, but several problems, including those caused by non-overlapping windows, remain to be resolved. However, the hierarchy definitely enhances the parallel extraction of textural features over increasingly larger receptive fields. Thus, while we believe that algorithms utilizing multiple levels of resolution are possible, further research in this direction has been put aside until a more complete development of our system has taken place.

III. Boundary Analysis

III.1 Introduction

One approach to the description of a scene is as a collection of boundaries hopefully delineating many of the objects (or parts of objects) in the scene. The boundaries themselves are usually composed of a collection of edges obtained by the application of a spatial differentiation operator to local neighborhoods in the digitized representation of the scene.[1] The edges represent local points of discontinuity in a particular feature, usually intensity, but possibly color or less often some textural feature. A variety of differentiation operators have been defined in the literature (for a review, see Davis [DAV75], and Rosenfeld and Kak [ROS76a].

The problems encountered with these operators are intrinsically related to the data upon which they operate as well as artifacts introduced by the operators themselves [RIS77]. The class of scenes being examined here produces edge data which is

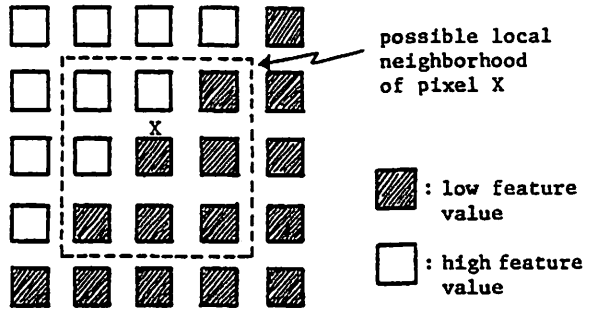
[1]Edge detection based on sequential tracking techniques will not be discussed here. It should be obvious that they are not well-suited to the class of images under consideration due to difficulties encountered in tracking through texture.

quite noisy. Boundaries very often are not described by step functions, but rather are formed by sometimes slow, and often irregular, changes in intensity. As a result, edge strengths are often distributed over an area as a function of lighting conditions, shadowing, surface properties, and shape. Variation in the form of texture produces high edge activity which is locally indistinguishable from "object" edges - indeed, in some cases an object boundary is not composed of well-defined edges at all [MAR75]. Thus, it should be evident that many decisions regarding boundaries cannot and should not be made purely at the local level, but instead should be delayed until contextual information can be brought to bear on the decision. Thus, the initial edge data should be as faithful to the original image data as possible, and implicit decisions based solely on local edge masks should be avoided [EHR78].

In this section, we first discuss the problems introduced by the choice of edge representation and operator, then the desirability of an interpixel representation of only horizontal and vertical edges, next an edge extraction analysis which allows the full horizontal and vertical components of a gradient boundary to be examined, and finally a relaxation process for updating edge probabilities based on their context. This process organizes the intensity data into locally consistent global boundaries.

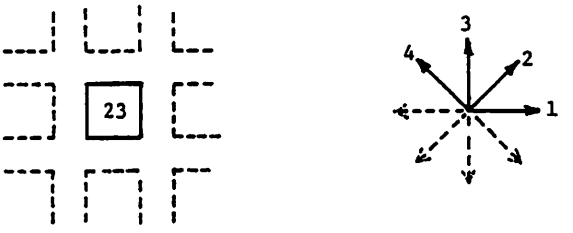
III.2 Considerations in the Development of an Edge Representation

The minimal amount of useful information obtainable from any spatial differentiation operator is an estimate of the magnitude of the intensity (or some other feature) change detectable over a local neighborhood, say the 3×3 window shown in Figure 4a. The resulting edge strength is often associated with the central pixel in the local window (assuming an odd window size). However, the magnitude of the gradient change provides no indication of the orientation of the edge, which might be crucial later when aggregating collections of edges into boundaries. Assuming that orientation is quantized to 45° intervals, any of the four



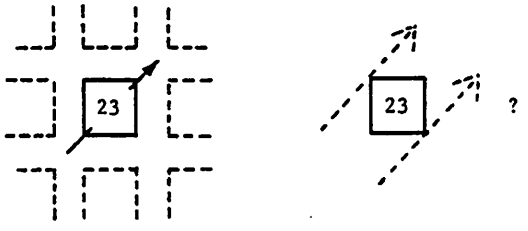
(a)

A portion of a digitized image as an array of pixels. For purposes of this discussion a 3×3 local neighborhood around a pixel is the local view assumed.



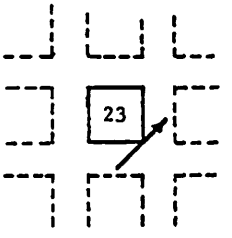
(b)

The magnitude of an edge in the vicinity of a pixel is assumed to be 23, but without orientation information there is ambiguity among the four orientations shown.



(c)

Even if the magnitude and orientation of a local edge is extracted, the placement of the edge relative to the pixel is still ambiguous.



(d)

An edge in a local neighborhood is unambiguously represented by a magnitude, orientation, and placement.

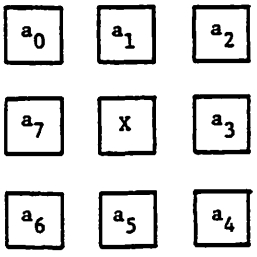
Figure 4. Considerations in the Development of an Edge Representation. Accurate placement of edges on an array of pixels demands that the magnitude, orientation, and placement of edges relative to the pixels be extracted and maintained.

directions shown in Figure 4b is possible. Therefore, let us assume our local operator determines the orientation of edges.

Even though the magnitude and orientation of the local edge is determined, there remains ambiguity in the placement of the edge (Figure 4c). We make the assumption that edges at a local level do not pass through a pixel, but rather partition the array of pixels into regions. It is usually the case that boundaries of surfaces in the physical world do not project onto a discrete array of digitized data exactly between pixels. However, local views of the data do not provide sufficient information to place the position and orientation of edges reliably on a continuous scale of image coordinates. We prefer to perform this placement after a more global boundary of many edges has been extracted. Therefore, an edge must be placed either to one side or the other of the pixel (Figure 4d). We conclude that an unambiguous placement and orientation of the gradient magnitude of a local edge is desirable.

Since the Kirsch spatial differentiation operator [KIR71,RIS77] is simple to compute and provides an estimate of both magnitude and direction, it appeared to be a good choice and we initially used it as our standard operator. It is defined on a 3×3 window, and for each pixel X it provides a local estimate of the edge contrast S(X), as well as edge orientation and placement D(X) (refer to Figure 5). This operator utilizes a fixed mask geometry at all orientations around a pixel; the best fit is associated with the submask whose output is maximum. Application of the operator at all pixel locations defines a transformation of the original feature data into an edge image. This implies an edge representation with eight possible orientation/placements of an edge with respect to each pixel as shown in Figure 5d. Examination of this figure yields several rather interesting observations which led us to abandon it entirely.

First, note that multiple, but logically equivalent, indications of each edge are possible. In Figure 6a diagonal edges d and e are associated with pixels X and Y, respectively, but a single edge through the window caused both. Clearly, it is desirable to have a unique indication of a single edge. This problem is resolved by adopting a canonical representation for the orientation/placement of edges, as shown in Figure 6b. Edges not in these standard positions can be shifted in a unique way to the pixel which has a logically equivalent edge in the canonical representation.



(a)

Local window for the Kirsch spatial differentiation operator.

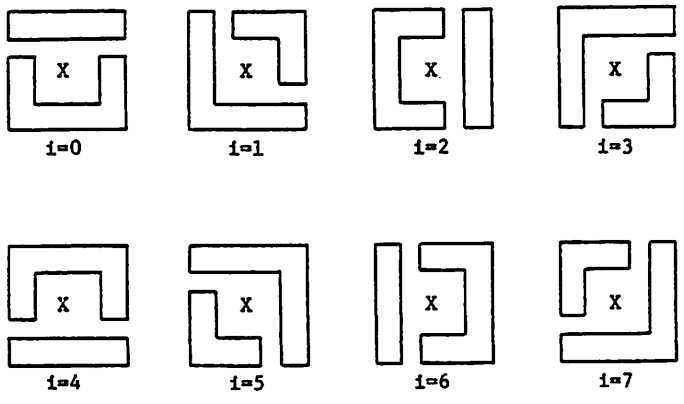
$$M_i = |5(a_i + a_{i+1} + a_{i+2}) - 3(a_{i+3} + a_{i+4} + a_{i+5} + a_{i+6} + a_{i+7})|; \; i = 0,1,\ldots,7$$

$$S(X) = \max M_i$$

$$D(X) = \{i | M_i \text{ is max}\}$$

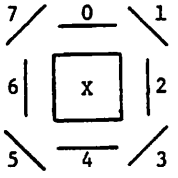
(b)

Definition of the Kirsch operator where indices are computed modulo 8. S(X) is the magnitude and D(X) is the orientation of the best edge associated with X.



(c)

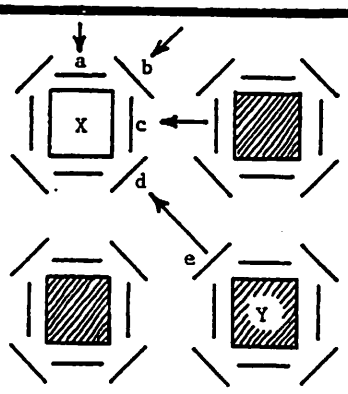
The eight submasks used; i is the index from (b).



(d)

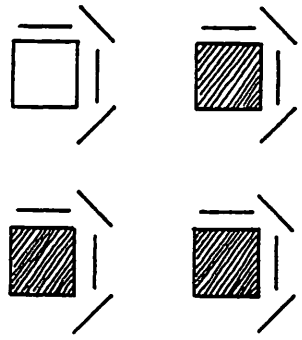
Alternative orientation/placements of an edge around X.

Figure 5. The Kirsch Operator. This operator provides the information required from an edge detector. It selects the strongest response from the set of eight alternative orientation/placements as the best edge.



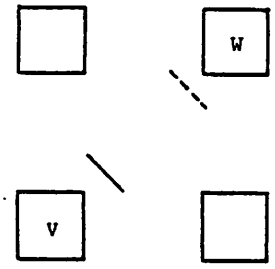
(a)

Possible orientations and placements of an edge. Note that the definition of the operator, applied to pixels X and Y, can lead to indications of edges d and e when only a single edge is present.



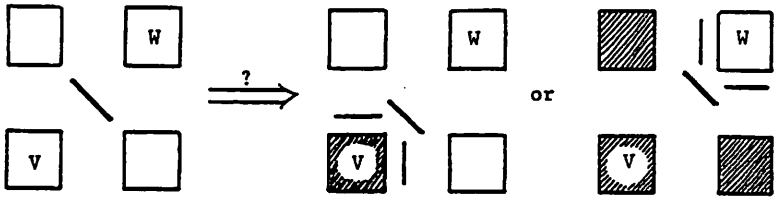
(b)

A canonical representation of orientation and placement of edges collapses multiple indications of an edge into a single response.



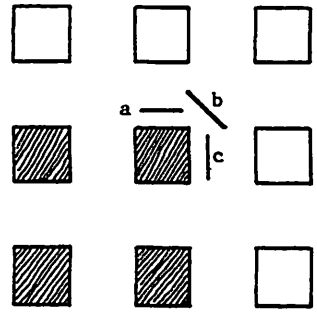
(c)

Canonical representation of the two possible placements of a diagonal edge between pixels V and W.



(d)

The presence of a diagonal between V and W is still ambiguous; the diagonal could be due to either corner of the type shown on the right.



(e)

There are eight edges competing for survival at each pixel. The existence of strong edges (a and c) is lost if only the single best edge associated with X is saved; the operator would represent the corner with the ambiguous diagonal edge b.

Figure 6. Further Development of an Edge Representation. Multiple indications of edges may be removed by a definition of a canonical orientation/placement, but diagonals remain ambiguous.

However, there are still 8 edges competing around a single pixel (or 4 edges if masks are associated only with their canonical positions).

Second, note that even with a canonical representation for diagonal edges, there is still inherent ambiguity in the partitioning of pixels. Figure 6c-d shows that if a diagonal edge between the pixels X and Y is not allowed to pass through the other pixels, then the representation of a diagonal, assuming no further information is extracted, is not sufficient to disambiguate between the two paths for this edge on the array of pixels (Figure 6d). If the diagonal edge is represented as a horizontal-vertical pair, then the local representation is disambiguated. This is an argument for delaying the decision about the exact orientation of a local edge until a later point when a more global view of a boundary is available.

Finally, the third problem with this representation compounds the previous problem. Note that only one of the 8 possible edges may be associated with the central pixel if a unique choice for the orientation/placement of the best edge is desired. Selection of the single best edge implies that potentially crucial information is being discarded; with eight edges competing for survival, the selection of the maximum contrast edge causes 7/8 of the edge information to be thrown away. In

Figure 6e edges a and c would be two of eight competing edges, and it is possible that only diagonal edge b at this orientation would survive; no indication of edges a and c would survive. Thus, this selection criterion allows events of potential interest to be discarded in favor of other events in the immediate vicinity.

The net result is a misrepresentation of the structure of the underlying information in that spurious orientations, odd gaps, and ambiguities appear in the edge data. One solution is to store more than one edge with a pixel. However, we have shown the redundancy that exists between diagonals and horizontal-vertical pairs. Maintaining multiple edges could lead to undue complexity in the organizational processes that are responsible for aggregating local edges into boundaries, because the interaction between adjacent edges is confused by the placement and orientation of redundant edges. The representation is not a clean one; we will not attempt to modify it any further and will discard it in favor of a simpler, more natural one.

The characteristics which edge operators and a good local representation of edges must exhibit in order to avoid the problems just described may be summarized as:

a) an edge should not pass through a pixel;
b) an edge should have an unambiguous location and orientation on the array of pixels;
c) there should not be multiple indications of a single local edge;
d) competition between edges should not cause a potentially interesting edge to be suppressed by another edge in the immediate vicinity;
e) the representation should lead to simplicity and clarity in the organizational processes for extracting global boundaries.

These constraints are satisfied by the choice of the interpixel representation of horizontal and vertical edges depicted in Figure 7 combined with the edge operator selected in the next section. In this representation the possible locations of edges are clearly defined and each edge has a unique position and orientation with respect to the pixels from which they were obtained. Thus, during application of the edge operator, edges of differing orientations do not have to compete for survival. This is a return to the representation in early work on region growing [BRI70] and used more recently in [YAK76,PRA77]. The final requirement, that it lead to clarity in organizing boundaries, will be the subject of the remainder of this section of the paper.

III.3 Problems with Large Edge Masks

There are still the problems arising from multiple edge indications from a single boundary. These effects can be traced directly to:

1) the size of the mask (or "window") over which the edge operator is defined; and
2) the spatial extent of the contrast change which forms the boundary (i.e., the gradient width).

If a boundary is defined by a step function, only a very small view is sufficient to see the entire change in contrast. On the other hand, small mask sizes will only have a partial view of boundaries defined by a gradient distributed over several (or many) pixels. This has led to approaches in which a hierarchical set of increasingly larger mask sizes are employed [ROS71,HAN74,MAR75,RIS77].

Let us examine the effect of using various mask sizes. One of the penalties incurred by using masks defined over large local neighborhoods is that responses will be obtained from masks placed at relatively large distances from the actual boundary. Given the interpixel representation, masks need to be applied at only two orientations: horizontal and vertical. Let us consider ideal boundaries which are represented by a step function between adjacent pixels. Figure 8 graphically illustrates the results obtained from several reasonable operators -- masks of size 1×2, 3×2, and 3×4 -- applied to an ideal image of a corner. It is easy to generalize these responses to conclude that all masks of size larger than 1×2 give rise to multiple indications of a single edge, thereby introducing a sometimes bewildering array of spurious edges at positions where none are actually present; in addition, edges at some of the correct positions have reduced responses for the larger masks. In these cases, the 1×2 edge mask is the only operator that will produce responses that are faithful to the underlying data. Given our concern about minimizing the distortion of data, we will only use the 1×2 mask, although the effect of this choice given boundaries defined by a wider gradient is not yet clear. We will show that use of the 1×2 mask is not a severe constraint and actually leads to an edge process which effectively is not limited to a fixed mask size.

III.4 Collecting the Gradient of Boundaries

As we have noted, any boundary which is not a step function across a pair of adjacent pixels will not have its total contrast detected by the 1×2

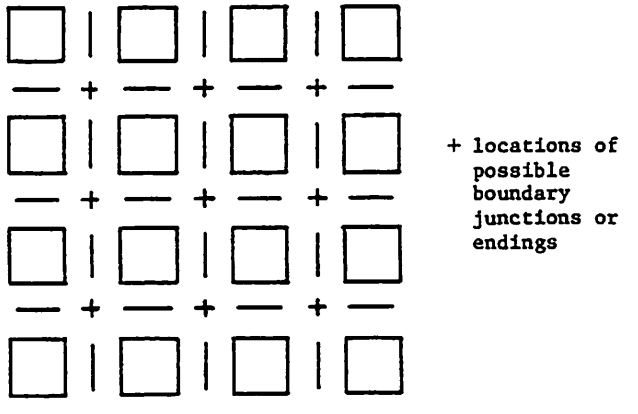
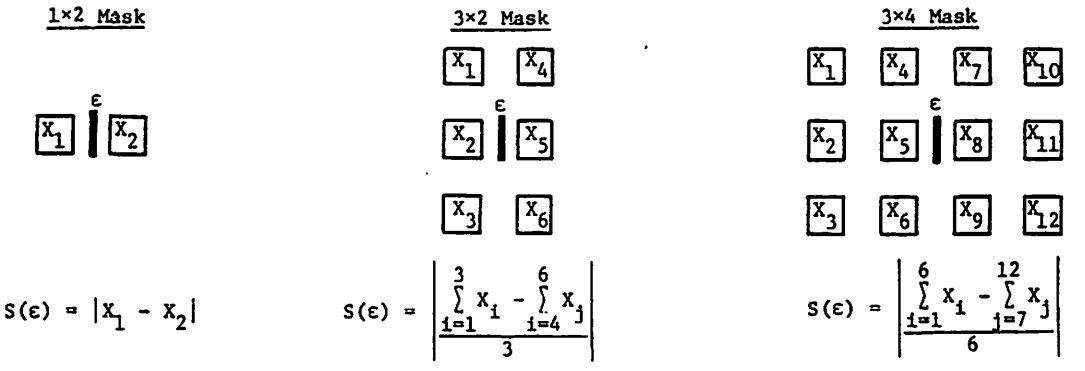
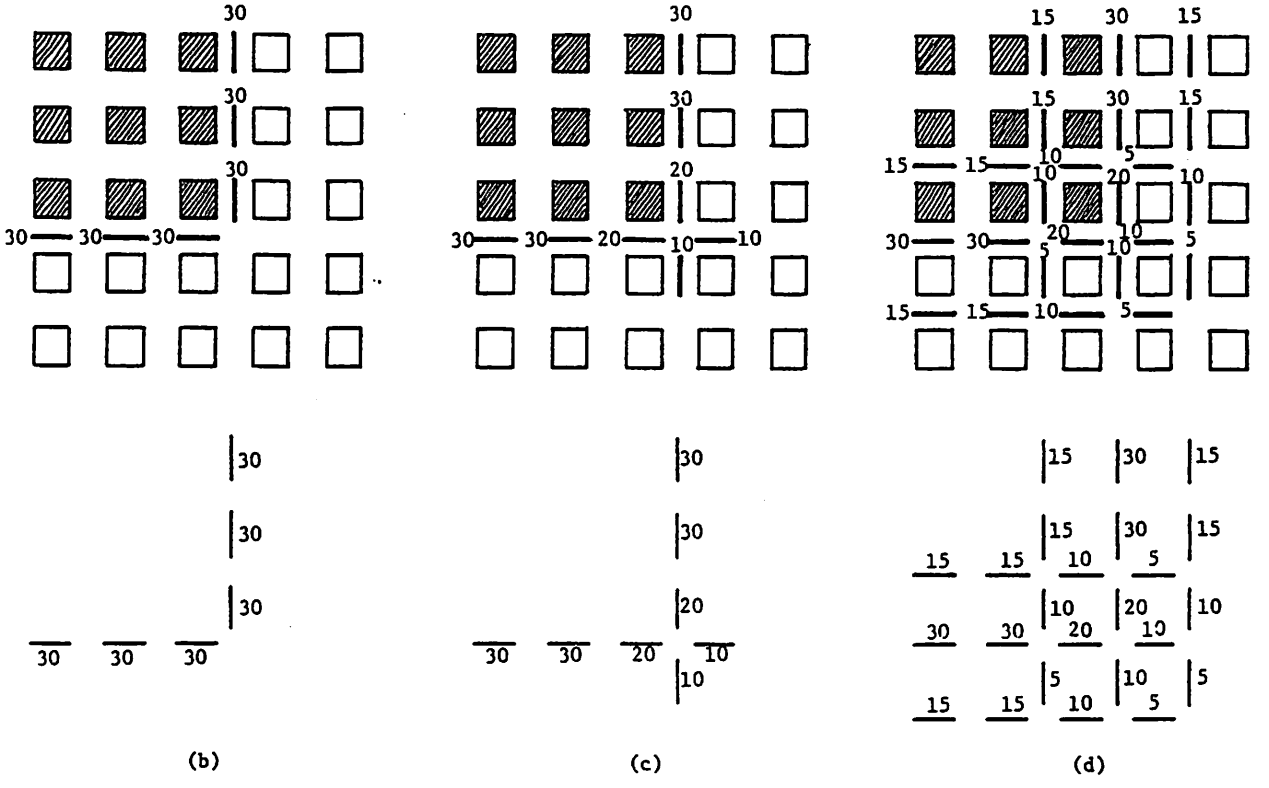


Figure 7. The Representation of Edges in VISIONS. The ambiguous diagonals have been removed leaving an interpixel representation of horizontal and vertical edges. This representation makes the placement and orientation of edges precise, as well as the location of possible boundary vertices (both junctions and terminations).



(a)

Definitions of edge masks of size 1×2, 3×2, and 3×4. The horizontal edge mask is obtained by a simple rotation of the vertical edge mask. S(ε) is the magnitude of the edge, orientation of the edge is implicit in the position of the edge, and the direction of the gradient change may be maintained by removing the absolute value (assuming appropriate care in the interpretation of the sign).



(b) Result of applying 1×2 mask.

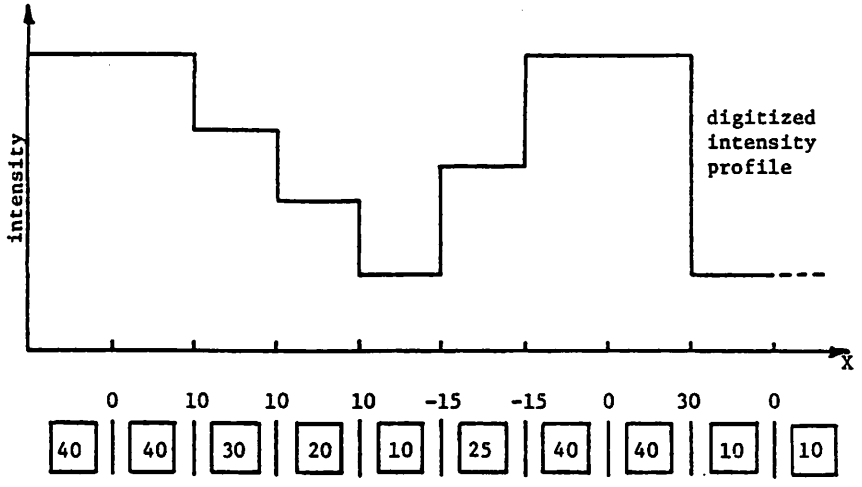
(c) Result of applying 3×2 mask.

(d) Result of applying 3×4 mask.

Figure 8. Comparison of Edge Operators. Examination of the results obtained after applying edge masks of varying size illustrates the difficulties encountered when larger masks are used. Note that in this figure cross-hatched pixels are assumed to have a brightness value of 10 and the others 40.

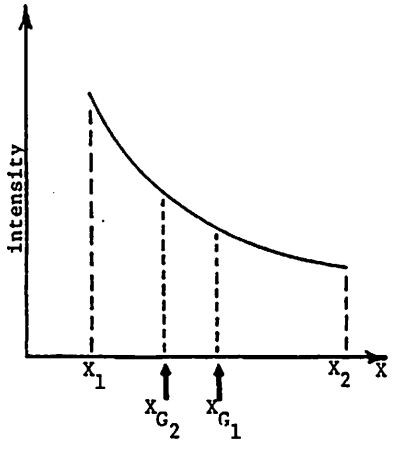
operator. Our solution to this problem is to extract the total contrast of a boundary by collecting the horizontal and vertical components of the gradient and to place the resultant value at a representative edge location.

For sake of discussion, let us consider only vertical edges, recognizing the symmetry involved. If the signed difference from the 1×2 operator is maintained, as depicted in Figure 9a, then the horizontal component of a gradient across several pixels is represented by a contiguous collection of vertical edges with the same sign; note that small local deviations of opposite sign or small plateaus (where there are no edges), could also be bridged. Therefore, it is straightforward to collect the total contrast into a single edge difference. Thus, the use of the 1×2 mask is actually a commitment to avoid the use of masks entirely; since the 1×2 operator preserves the topology of the intensity scan line [EHR78] (up to a constant), the effective mask size is easily determined dynamically by the actual structure of the data. The magnitude, width, and position of the undistorted gradient is available in the local edge data.



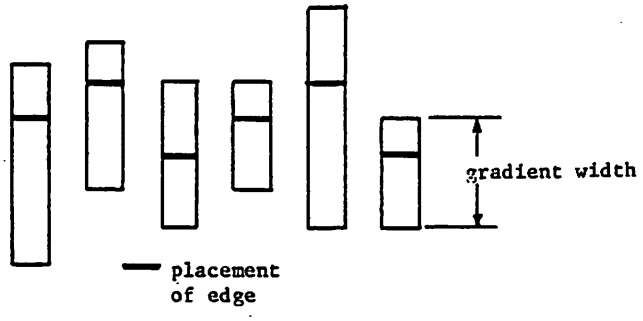
(a)

Application of the 1×2 edge mask to a sample horizontal intensity scan line. The topology of the scan line is preserved by the 1×2 mask and may be reconstructed from the signed edge values up to a constant. In effect the 1×2 mask is not really a mask in the usual sense; it is a measure of the local horizontal (or vertical) component of the gradient.



(b)

Alternative strategies for placement of the collected magnitude of the gradient include the midpoint X_{G_1} of the gradient, and the position X_{G_2} where the first moment about that point is zero.



(c)

No matter what method of edge placement is used, misalignment of edges from one scan line to the next, due to local changes in the width and topology of the gradient, must be expected.

(d)

A gradient may not be strictly monotonic; a more global context-sensitive decision concerning the extent of the gradient should be made before the edge defined by the gradient can be determined.

Figure 9. The Gradient Collection Process. The 1×2 mask has only a local view of wider gradients. More global contrast across gradients must be extracted.

There are a variety of strategies for placing the collected magnitude as a (set of) positions(s) as shown in Figure 9b. It could be placed at the geometric center of the gradient width. The one used in this paper is to place the value at the interpixel location closest to the center of gravity of the gradient (i.e., where the first moment is zero). Due to local variations in adjacent scan lines, this can lead to placement of adjacent edges which are misaligned as shown in Figure 9c. Recall that these initial local analyses are to form the input to an edge aggregation process which will be responsible for globally organizing the set of edges. Therefore, in order to avoid errors, a function of the total constrast of a boundary will be placed at several potentially "good" positions around the center of gravity. The final value assigned to an edge location in a gradient is the total contrast detected over the gradient scaled down by its distance from the center of gravity of the gradient. All other edges in the boundary are suppressed to some small non-zero value. Thus, the potential for positioning errors is explicitly recognized, and final decisions regarding edge placement and boundary organization is left to the context-sensitive relaxation processes to be described shortly.

There are additional problems in that the one-dimensional gradient of a boundary is not always monotonic (Figure 9d), and might require a more complex context-sensitive examination of the scan line in the gradient collection process [EHR78]. Ultimately, it ought to be possible to utilize a gradient collection process which varies the particular model of gradient activity under the direction of semantic processing. This model could be obtained from a higher level shape processor, for example, which may predict a particular gradient model based on shape attributes (e.g., expected shading given a light source and 3D surface properties). Such a system would initially utilize a simple gradient model and then use the predicted model to further refine edge placement.

III.5 Organizing Edges into Boundaries Via Relaxation Processes

The information contained in the edge representation is still unorganized in that the edges have been obtained from local processes and have not yet utilized contextual information to organize them into boundary segments (sometimes referred to as lines) and vertices. A powerful method for accomplishing this is based on a paradigm of local cooperation and competition between edges; that is, a local edge should be viewed in the context of surrounding edges, and decisions regarding edge properties (including existence) should be delayed until the context is clear. By suitably defining a local neighborhood of an edge, the edge activity in that neighborhood may either support the central edge or deny its existence. By overlapping the neighborhood and iterating the decision process, local effects can propagate and affect surrounding neighborhoods. For each iteration, the totality of effects in the local context will be used to update the probability of the existence of each edge. Thus, the approach taken here is the definition of a local neighborhood, the definition of a set of contextual "events" in the neighborhood of the edge, and the embedding of these definitions in a uniform iterative procedure, called a relaxation process [ROS76b], for performing the updating of the edge probabilities.

The smallest meaningful neighborhood for an edge in our representation (Figure 10) must include the three edges to either side where continuation of that edge could occur as a line; it must also include the two parallel edges which are possible alternative placements for the collected gradients. Note that the (signed) probability of an edge may be obtained by normalizing the absolute value of the signed edge strengths with the maximum absolute value after gradient collection.[1] The sign will be maintained with this probability as an encoding of the direction of contrast change.

Now let us provide a brief review of the key ideas of relaxation processes; for a more complete discussion see [ROS76b,ZUC76,RIS77]. The general idea is to compute some probability updating contribution Δ for the central edge as a function of the probability of the neighboring edges. It is assumed that each edge location has a set of N possible labels $\{\lambda_1,\ldots,\lambda n\}$ which can be associated with it (e.g., in other applications n labels have been the n-1 alternative orientations for the edge and a label for the absence of an edge). We will use $P_i(\lambda_k)$ to denote the probability of label k at the i^{th} edge location, LOC_i. Furthermore, it is assumed that there is some means for computing a reasonable initial probability for each label at each edge location. Then each label at each LOC_j contained in the neighborhood N_i of LOC_i will be used to update $P_i(\lambda_k)$, $k = 1,\ldots,n$. $P_i(\lambda_k)$ will be increased (decreased) by label λ_m at LOC_j if the labels are compatible (incompatible) where the effect of this change is weighted by $P_j(\lambda_m)$.

[1] A more effective procedure is to normalize the strength of an edge by the maximum strength edge over a large, but local, neighborhood. Otherwise a single very strong edge anywhere in the image could reduce the probability of edges to a small value everywhere in the image.

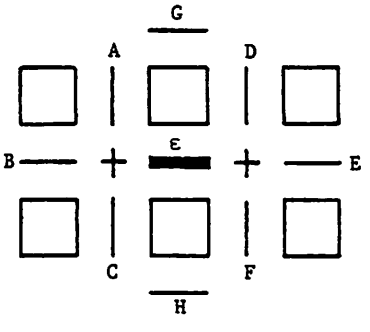


Figure 10. The Local Neighborhood of an Edge. ε is the central edge to be updated as a function of the edges in the neighborhood. The set of edges {A,B,C} and {D,E,F} represent possible continuations of ε to each of its sides; parallel edges G and H are potential competitors for ε; + represents possible vertex locations.

Compatibility is defined in terms of a function r_{ij}:

$r_{ij}(\lambda_k,\lambda_m) > 0$ if λ_k and λ_m are compatible,
< 0 if λ_k and λ_m are incompatible,
$= 0$ if λ_k and λ_m are independent.

Then, $\Delta P_i(\lambda_k) = \sum_{j\in N_i} d_{ij} \sum_{m=1}^{n} r_{ij}(\lambda_k,\lambda_m)P_j(\lambda_m)$, where d_{ij} is a weighting of the influence of LOC_j upon LOC_i and keeps ΔP_i in the interval from -1 to $+1$. Denoting the probability of label λ_k after the t^{th} iteration as $P_i^{(t)}(\lambda_k)$, it will be updated as follows:

$$P_i^{t+1}(\lambda_k) = \frac{P_i^t(\lambda_k)[1+\Delta P_i^t(\lambda_k)]}{\sum_{k=1}^{n}[P_i^t(\lambda_k)(1+\Delta P_i^t(\lambda_k)]}.$$

Note that the denominator is a normalizing factor computed across the new probabilities of the n labels, so that the new values for P_i^{t+1} will sum to one.

The interpixel edge representation we have chosen again provides simplification. There are only two labels at each location, EDGE and NOEDGE. Only a single probability, $P_i(EDGE)$, is necessary for determining the probability of both labels since $P_i(NOEDGE) = 1-P_i(EDGE)$. In effect the updating process is only a function of the probabilities of EDGEs in the neighborhood. Now we will make one more simplifying assumption, that it will only be necessary to compute $\Delta P_i^t(EDGE)$, and that $\Delta P_i^t(NOEDGE) = 0$. The effect of this assumption is that it is sufficient to allow the likelihood of NOEDGE to vary inversely with the likelihood of EDGE. Now we can use P_i^t to represent $P_i^t(EDGE)$. The computation of P_i^{t+1} follows the general equation

$$P_i^{t+1} = \frac{P_i^t[1+\Delta P_i^t]}{P_i^t[1+\Delta P_i^t]+(1-P_i^t)}$$

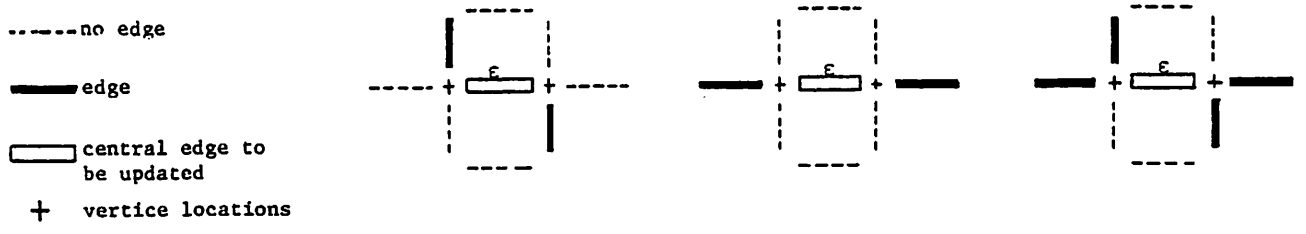
and simplifies to

$$P_i^{t+1} = P_i^t\left[\frac{1+\Delta P_i^t}{1+P_i^t\Delta P_i^t}\right]$$

III.6 Labels in a Local Context are not Independent

Now we must specify the mechanisms for computing the updating factor Δ. One might try to specify meaningful compatability coefficients [ZUC76] for each edge in the defined neighborhood (Figure 10). For example there ought to be positive compatability coefficients on edges at the six locations representing possible continuations of ε (three to each side), whereas there ought to be negative coefficients for edges at locations of parallel edges G and H with the same gradient sign. Note that we cannot use a negative coefficient for the absence of an edge at locations A, B, or C, for example, because the absence of one of these should not inhibit ε -- one of the other two might be present forming a path of good line continuation. As we shall show next, this is a distinct limitation of the relaxation process as it is formulated in the previous section.

As the updating process has been specified, the two labels at each edge location in the neighborhood of ε would be used independently to contribute to Δ. The following notation for edges and the categorization of the junctions to either side is very useful for further analysis and we follow [PRA77] in this regard. Figure 11a describes the notation used for



(a) Notation.

(b) Two situations for which the probability of the central edge, P_ε, should be increased as a function of the neighborhood.

(c) A situation for which P_ε should not be increased because ε is not essential for good line continuation to the left or right (refer to Figure 13).

Figure 11. The relaxation updating process is often restricted to a linear function of the labels on the edges in the local neighborhood, where compatability coefficients act as variable weights. There is no set of compatability coefficients for a linear function which leads to the desired updating in all the contexts which can occur.

the presence and absence of edges in the figures that follow. Figure 11b portrays two cases where ε should be increased, whereas Figure 11c portrays a case where ε should not be increased because it is not necessary for good line continuation (the basis for these decisions will be discussed further in the next section). Since the last situation is a linear combination of the other two, there is no linear combination of weights to satisfy these requirements. The independent use of labels prevents many events of interest (which are entirely observable within the neighborhood of ε) from influencing ε. Since the events cannot be expressed as a linear combination of the labels, they cannot be specified as separate events; n-ary relations on the labels are required.

This limitation may be overcome by generalizing the notion of a label to that of a pattern defined over the neighborhood. The probability of a pattern is expressed as a (potentially) non-linear function of the probabilities of the 8 edges in the neighborhood. The compatability coefficient is now between the i^{th} pattern of neighborhood labels and the labels of ε, not between labels of the j^{th} neighboring edge and the labels of ε. Thus, r_{ij} is replaced by a weight (positive or negative) which determines the effect that the pattern should have on ε. (Of course this approach could be generalized to larger neighborhoods.) The total contribution Δ is then defined as a linearly weighted sum of the probability of the n patterns:

$$\Delta^t = \sum_{i=1}^{n} \Delta_i^t = \sum_{i=1}^{n} W_i P_i^t$$

This generalization of relaxation bears some relationship to production systems in which each production rule is specified as a pattern for triggering the rule, and a response form to be executed once the rule has been triggered. Each of our patterns can be thought of as a rule with some probability of being present -- in other words, all patterns are simultaneously true to some degree. As the effect of some patterns propagate and begin to organize local contexts, other patterns become true to a different degree. Each edge responds to changes in its context, but in places where there is ambiguity, the updating often has competing influences causing organization in the local context to be delayed. In this way influence will spread from islands of unambiguous contexts within the image. If this dynamic process is to have a meaningful result, it is incumbent upon the designer of the patterns to capture the syntax/semantics of line boundaries (e.g., laws of good gestalt) in the local contextual patterns.

It is worth noting that initially an edge with probability P on the array of pixels must have some non-zero continuation on both of its ends. In fact among its three possible continuations, the minimum possible value of $MAX\{P_A, P_B, P_C\}$ is P/3 because the pixels which produced the evidence of this edge are related to the other three edges.[1] As the edges around a vertex are crossed in a clockwise or counter-clockwise direction, the signed differences must sum to zero. A simple illustrative example appears in Figure 12.

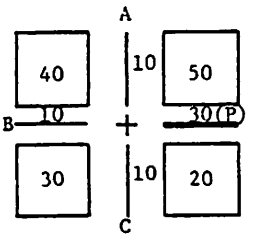


Figure 12. Constraints on the Edges in the Neighborhood of an Edge. An edge of some strength P on a square array of pixels must have evidence of continuation no less than P/3 on at least one of the three edges to each of its sides.

III.7 Contextual Patterns for Updating Edge Probabilities

It only remains now to define the patterns over the local neighborhood of an edge. Since the goal of the analysis is to extract continuous boundary segments, the patterns should focus on those conditions which support or deny the hypotheses that the central edge is part of a more global boundary.

The central edge ε has junctions at both of its ends. If edge ε is ignored for the moment[2], each junction can be viewed as a vertex of some degree between 0 and 3; the classification of vertex types is summarized in Figure 13a. Let us use the notation i-j to denote the case where the pair of vertices has degree i and j respectively. Subject to symmetry i-j≡j-i, and only i-j vertices where i≤j will be referred to. Also, for this treatment, a configuration of n edges will be considered an equivalence class regardless of their particular location in the three possible locations. Then, only cases 0-0 through 3-3 shown in Figure 13 [PRA77] and parallel edges G and H need be considered.

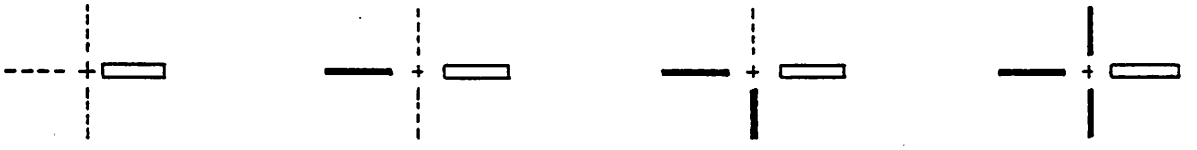
Before discussing these cases in detail, let us first briefly summarize the edge semantics associated with equivalence classes of these cases:

[1] Note that if the edges are represented on a discrete scale that this condition is true up to the integer error at the lowest levels of numeric resolution. Also note that the effects of gradient collection can cause this condition to be violated by inconsistent placement of collected edges.

[2] Note that when ε is also considered (as will be true after the edge updating process is complete), a vertex can have degree between 0 and 4. A vertex of degree 1 represents a line termination, of degree 2 a line continuation, and of degrees 3 and 4 a line intersection.

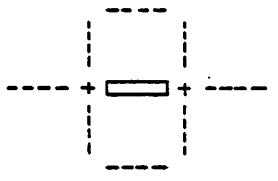
1) {0-0} is an "isolated edge" and ε should have negative support;
2) {0-2,0-3} is a "spur" and ε should have negative support;
3) boundary continuation
   3.1) {0-1} is an "uncertain boundary continuation" or "boundary termination" and ε will have weak positive support;
   3.2) {1-1} is "certain boundary continuation" and ε will have strong positive support;
   3.3) {1-2,1-3} is "continuation to a boundary junction" and ε will have medium positive support;
4) {2-2,2-3,3-3} is an ambiguous "bridge between boundaries" which is potentially unnecessary, and will have no effect upon ε;
5) {edges G,H|sgn(G)=sgn(ε),sgn(H)=sgn(ε)} are "competing parallel edges" and will have a negative effect upon ε.

Figure 13b is the 0-0 case where ε is an isolated edge and should be inhibited (i.e., in the absence of any additional information, P_ε should be reduced). Figure 13c,d, cases 0-2 and 0-3, are both "spurs," where ε has no continuation to one side while there is a line continuation or

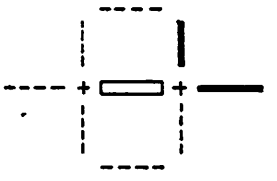


(a)

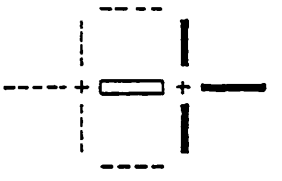
Classification of vertex types. The degree of a vertex is defined to be the number of edges at one end of the central edge (but not including the central edge). From left to right they are Type 0, 1, 2, and 3 vertices, respectively.



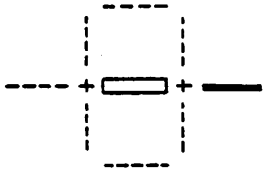
(b) 0-0: Isolated edge.



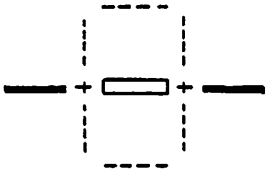
(c) 0-2: Spur.



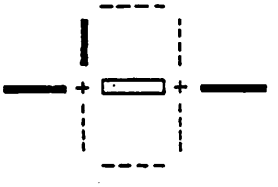
(d) 0-3: Spur.



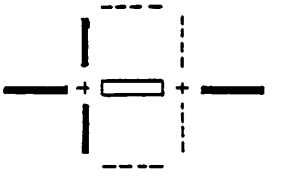
(e) 0-1: Uncertain continuation or line termination.



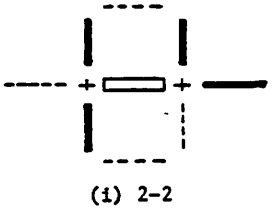
(f) 1-1: Boundary continuation.



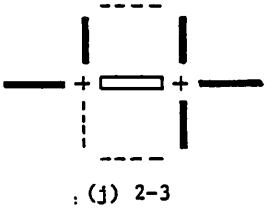
(g) 1-2: Continuation to junction.



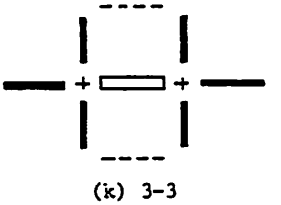
(h) 1-3: Continuation to junction.



(i) 2-2



(j) 2-3



(k) 3-3

(i-k) Bridge between boundaries.

Figure 13. Equivalence Classes of Vertex Pairs. i-j≡j-i represents a vertex pair of degree i and j, respectively. Ignoring parallel edges G and H in Figure 10, there are only 10 cases, which form 4 major contextual events (one of which is divided into 3 subevents).

line intersection on the other side; here we also inhibit ε so that unnecessary spurious lines do not grow out from many places on a continuing boundary, or at a junction of lines.

Figure 13e represents the 0-1 case where a line terminates and its continuation is uncertain. It has been argued [PRA77] that no updating should take place here because it is not dependent upon the local neighborhood, but a wider context instead; for example if there is a similar line termination close by, it is probably desirable to grow the boundaries together. If the size of the local neighborhood is increased, the number of patterns to be examined increases sharply, with an associated sharp increase in computation. Therefore, the analysis will be kept simplified by constraining it to the small neighborhood defined in Figure 10. We allow the edge ε in the 0-1 case to be weakly supported, providing slow growth of the boundary. Note that with a line termination, all three edges which are absent will receive this support, and the line will grow in all directions seeking another boundary to which it can anchor. This problem can be alleviated by allowing the original data to guide this growth so that the entire process is anchored to "reality." This will be examined in more detail elsewhere [KOH78].

The desirability of supporting ε is clearer in the 1-1, 1-2, and 1-3 cases shown in Figure 13f-h. The clearest case is for the 1-1 case, where ε is necessary for good line continuation. Its absence would form a discontinuity in the boundary and the two edges nearby each other would become boundary terminations. In the 1-2 and 1-3 contexts, the central edge ε is necessary only for continuation of the single edge associated with the degree 1 vertex. Thus, the presence of edge ε is only necessary for good continuation of one edge in the 1-2 and 1-3 contexts, while it is necessary for two edges in the 1-1 context. Consequently, support of ε from the 1-1 context can be weighted more heavily. Also note the difference between the negative effect of the 0-2 (0-3) case and the positive effect of the 1-2 (1-3) case. An edge hanging off a line continuation is a "spur" if it has no other edge with which to link -- if there is no local evidence of a boundary to one of the sides of ε. On the other hand, the one edge in the 1-2 case could represent a left boundary which is to link to the right boundary for which there is already evidence (i.e., the degree 2 vertex).

The last three local vertex pairs, 2-2, 2-3, and 3-3, (Figure 13i-k) represent cases where there is evidence of two lines passing nearby each other. The central edge ε is not a function of these lines because its presence or absence does not affect good line continuation for the edges which are present, and therefore the central edge ε will not be inhibited or supported, but instead left alone. To the extent that the patterns of one of these last cases is actually present (in terms of probability), all of the other patterns we have discussed are not present, allowing the updating process to decouple. In other words, the probability that cases b to h are present is equal to the complement of the probability that cases i to k are present. The effect is that ε will have some initial probability based upon the image data, and then it will be updated (the probability increased or decreased) by the patterns described until the degree 2 vertices have probability one to either side. Thus, P_ε will be updated via the context, and if the context clearly forms itself into a 2-2, 2-3, or 3-3 vertex pair, edge ε will no longer change. Its value will remain stable at the last value it had before the neighboring edges reached probability 1. This will happen over a number of iterations and further analysis of the situation will be left for post-processing.

Now we must specify the computation of the probability of each of the patterns in Figure 13 in terms of the probabilities of the edges in the local neighborhood. We emphasize again that each of the cases shown can simultaneously have some non-zero probability. If the probability of a vertex of degree i, i=0,...,3, is defined, then the probability of an i-j vertex pair will be the product of the probability of each of the two vertices. Since the semantics of a vertex of degree 3 is identical to those of a vertex of degree 2 in all the cases described, then only a vertex of degree at least 2 will be computed to represent both. For a reduction in computation, the probabilities of vertices of degree i will be approximated as a function of the three edges A, B, and C as follows:

$$P(\text{vertex of degree 0}) = 1-\text{MAX}(P_A,P_B,P_C)$$
$$P(\text{vertex of degree 1}) = \text{MAX}(P_A,P_B,P_C) * [1-\text{MAX2}(P_A,P_B,P_C)]$$
$$P(\text{vertex of degree at least 2}) = \text{MAX}(P_A,P_B,P_C) * \text{MAX2}(P_A,P_B,P_C)$$

where MAX is the largest of the values and MAX2 is the second largest of these values. Due to the approximation of the probabilities of vertices, instead of exact computation, they will not sum to one; this is not important because they are only used to compute relative changes in the label probabilities which are then renormalized. Finally, note that the probability of both an i-j vertex and a j-i vertex must be used to update ε.

There is one more condition that must be considered: parallel edge suppression for edges with the same gradient sign. At the beginning of the process such edges will be present throughout the image because of the multiple placement of edges during gradient collection. These then compete for the honor of representing the boundary. The probability of this condition will be defined to be the maximum of the edges G and H which have the same gradient sign as ε; if neither do, the probability of this pattern will be set to zero.

III.8 Results

The results of this process are shown in the following series of figures. Figure 14a shows a black and white intensity image produced by averaging over the three color components. The image, differentiated using the 1×2 operator, is shown in Figure 14b, where the brightness of edges encodes the strength of the edge. Figure 14c shows the gradient collected image which has been normalized to a zero-one range by the maximum strength edge, brightness now encoding the edge probability. The results after iterations, 1, 2, 5, 10, and 20 are shown in Figure 14d-h. The results are quite effective and capture most of the local variations

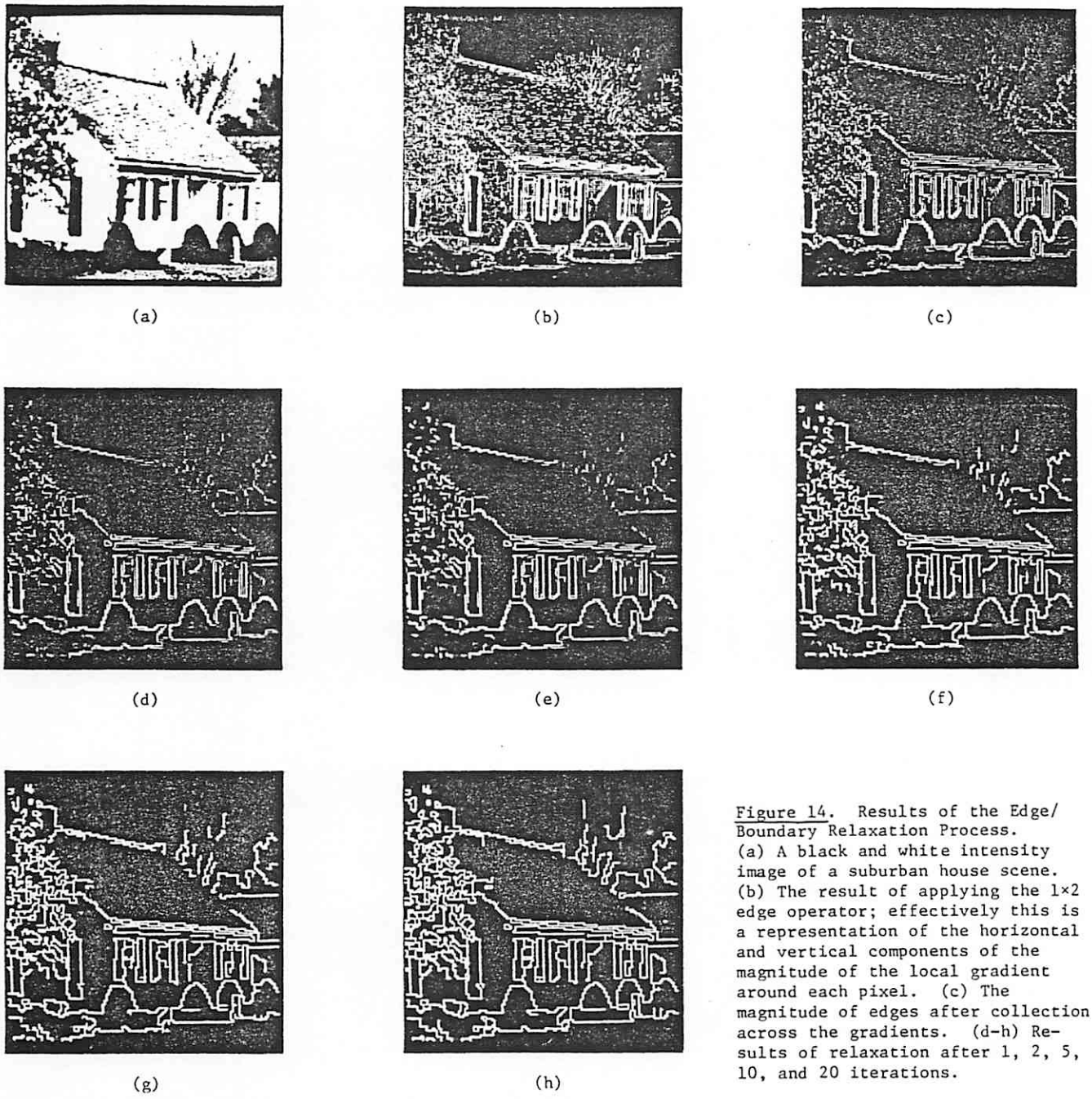


Figure 14. Results of the Edge/Boundary Relaxation Process. (a) A black and white intensity image of a suburban house scene. (b) The result of applying the 1×2 edge operator; effectively this is a representation of the horizontal and vertical components of the magnitude of the local gradient around each pixel. (c) The magnitude of edges after collection across the gradients. (d-h) Results of relaxation after 1, 2, 5, 10, and 20 iterations.

within the image.

The weight associated with the isolated edge case (0-0) can be used as a boundary sensitivity control. As this negative weight is increased, the edges which organize into boundaries and then support each other must be stronger in order to grow in the face of this increased suppression. This weight acts as a negative umbrella across all edges. Thus, stronger or weaker boundaries can be extracted by varying this value. The boundaries extracted for an increased value of this weight are shown in Figure 15. Note that only the stronger edges of Figure 14 survive to organize into boundaries. Figure 16 shows the results of the analysis on a different image.

We believe that the approach described here represents a significant improvement over earlier approaches to boundary/edge analysis. The decoupling of effects of representation, differentiation operators, gradient analyses, and aggregation processes leads to a clear delineation of the requirements of each process. The representation allows an extremely clear analysis of edges in relation to pixels and could potentially lead to a comprehensive theory of edges and pixels.

IV. Region Formation

The boundary formation algorithm described in the previous section involved the aggregation of edge information based on local spatial views of

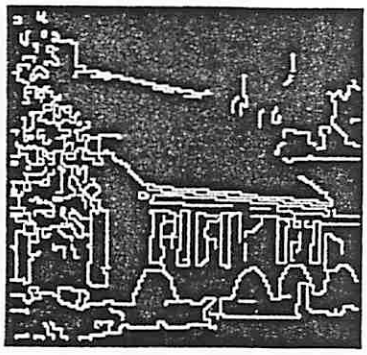


Figure 15. Varying the sensitivity to edge/boundary strength. Increasing the weight associated with isolated edges results in only (relatively) stronger boundary segments surviving. Similarly, decreasing this weight allows weaker segments to survive. The results shown are after 10 iterations, using the same parameters as for Figure 14 except for decreased sensitivity via the weight on isolated edges; weaker boundary segments do not survive.

the data. In this representation of the image data, all local variations are potentially important. Therefore, in textured areas, the boundaries produced by this algorithm tend to delineate reasonably small homogeneous regions in the image, such as the patchwork of boundaries in the tree region on the left side of Figure 14.

Our goal in the formation of regions is to organize the image on the basis of a more global view of the data. We wish to suppress the local textural variations in an attempt to produce a representation of the image in terms of a relatively small number of major regions with associated attribute descriptors. One of the standard techniques in the literature involves the analysis of cluster activity in global histograms. However, there are serious difficulties with these approaches in general scenes having a significant degree of textural complexity [RIS77].

In this section we will first outline the transformation of color data to produce effective color features. Then we will examine the fundamental problems in the standard use of histograms: they do not provide an effective linkage between image space and feature space. This linkage may be provided by defining a mapping from extracted clusters in histogram space to probabilities in image space, where the probabilities reflect the degree to which an image point belongs to each of the clusters. This permits a straightforward relaxation process to be defined for the spatial organization of regions. Results are presented to support the effectiveness of this process. Finally, we describe a hierarchical representation for segmentation results.

IV.1 Color Feature Space

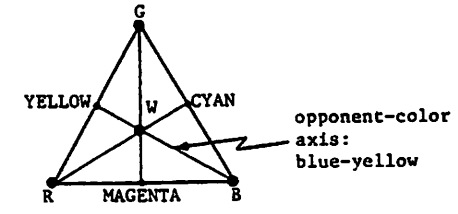
The histogram analysis techniques depend on the measurement of some feature(s) of the image points, possibly including those originally used to represent the actual scene. For color images, the usual features initially measured are the red, green, and blue components (RGB) of the light level at each point in the scene. From this information, a variety of other representations, such as normalized RGB, or hue, saturation, and intensity (HSI), may be derived [TEN74,RIS77]; because many of these transformations are nonlinear, they give rise to distributions with unavoidable singularities [KEN76]. The presence of these singularities may severely complicate analysis of the resulting histogram. In order to avoid these difficulties, it has been suggested that analysis be restricted to linear transformations of RGB, such as the YIQ representation used in the television industry.

(a) (b) (c)

Figure 16. Results of the Edge/Boundary Relaxation Process. (a) 1/2 differentiation operator applied to an intensity image of a farmhouse scene (Figure 24a). (b-c) Results of relaxation process after 2 and 20 iterations.

More recently, Sloan and Bajcsy [SLO75] have argued for the use of an opponent-color representation which has been proposed as underlying the color mechanisms in human vision [COR70]. Simply stated, the effect of this transformation is to parameterize the RGB color data into an equivalent set of features which have particular complementary colors at the extremes of their scales; for example, a feature whose opponents are blue and yellow would provide information on the relative amounts of blue and yellow present. The "zero" point in the scale, where equal amounts of each hue are present, is white.

Figure 17 illustrates a very simple linear computation of opponent color features. Figure 17a is a standard way of depicting color information on a triangle, where the most saturated possible values of R, G, and B are associated with the vertices. A point interior to the triangle represents a color which can be obtained by combining specific amounts of the R, G, and B primaries; points on the perimeter are totally saturated while interior points are less saturated (i.e., diluted by white light). The interior point W, equidistant from the vertices, represents white light composed of equal amounts of R, G, and B. It forms a neutral gray, including black and white.

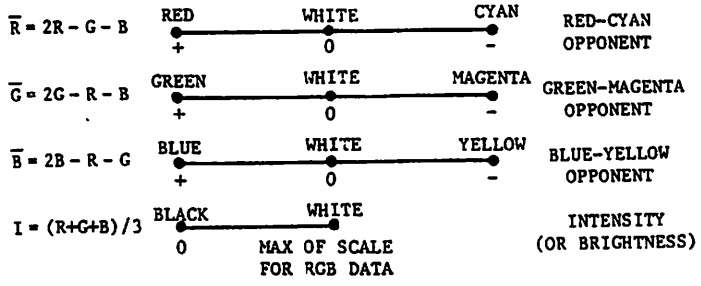


(BLACK ≡ GRAY ≡ WHITE)

(W : WHITE ≡ GRAY ≡ BLACK)

(a)

The color triangle. The axes passing through the neutral gray point represent the opponent-color features.



(b)

Opponent color features are approximated as a linear function of the RGB data. They provide a way for assessing actual color as a scalar feature in a more meaningful way.

Figure 17. Computation of Opponent-Color Features

Each of the three axes shown in Figure 17a represents an opponent-color feature and is uniquely determined by the line from each vertex passing through W. These opponent-color features will be approximated by the linear functions shown in Figure 17b, with a constant added to each feature in order to obtain a range of positive values. The features will be referred to as $\overline{R}$, $\overline{G}$, and $\overline{B}$, but the reader should remember that the letter used represents only one end of the feature. These features are an approximation to the opponent features. As the computation of $\overline{R} = 2R - G - B$ illustrates there are many values of G and B which produce the same value for $\overline{R}$, although the feature is an accurate measure of cyan only when G and B have equal values. The reader should also note that $-\overline{R} = \overline{G} + \overline{B}$, and therefore only two of the three features are independent. By adding an intensity feature, computed as the average of the R, G, and B components, all the information in the original RGB data is preserved. Also note that the axes are not orthogonal to each other; $\overline{R}$, for example, can be expressed as $\overline{R} = \frac{1}{2}[3(R - B) - \overline{G}]$. This dependency leads to the orderly spacing of $\overline{R}$ and $\overline{G}$ components (for example) when two-dimensional histograms are computed (refer to Figure 23).

If the axes are not constrained to go through W, there are an infinite number of possible opponent-color features. It should be possible to tailor the features actually used, via feedback from semantic processes, to provide the greatest discrimination between particular objects of interest.

While only three of the four features described are independent, in a particular circumstance any may be useful. Consequently, all four features plus the three RGB features will be utilized as point properties (as opposed to textural features) of the data.

IV.2 Image Space, Feature Space and the Failure of Histogram Analysis

A number of the techniques in scene analysis commonly used for organizing global visual information are based on histogram analysis. Here, the values of some feature (e.g., intensity) across the image are counted and the resulting distribution is examined. A classic technique has been to make use of "clusters" in the distribution to set a threshold on the feature values [OHL75,NAG77,ROS77a,b]. Each image point is then labelled according to the histogram cluster of which it is a member. A light object can be separated from a dark background (or vice-versa) by a proper setting of this threshold. However, in images with some degree of textural complexity, such a result cannot be expected. The distributions of the individual goal regions can overlap and clusters from regions can be obscured.

A more recent development in histogram analysis [OHL75] is the recursive iterative examination of one-dimensional histograms associated with a set of features. The clearest cluster in the set of histograms is extracted and used to "turn on" associated points in the image. The hope is that a large subset of the points contributing to the cluster will come from a spatially connected set of image points; this would allow the region to be extracted using simple techniques. By recursively histogramming the subregions which are formed (as well as the areas of the image which remain), the analysis attempts to isolate and remove the largest

regions so that the presence of less noticeable peaks will become obvious. Textured areas are extracted by a "busyness" measure based on the density of edges. The recursive analysis can be quite effective, because each resegmentation potentially enhances the effectiveness of other features.

A variation of this technique employing 2D histograms for a pair of features has been employed [HAN75] with the histogram itself analyzed in the processing cone [HAN74]. More recently this has led to techniques for gross histogram clustering in order to develop region plans [NAG77] (more on this in Section IV.8).

These approaches have had dramatic success for some images. However, there are very serious difficulties. Although histograms provide a global view of the feature data, they do not reflect the spatial information in the image from which they were derived. Figure 18 illustrates the type of problems that arise. Assuming the core of clusters C_1 and C_2 are obvious and easy to find, there is ambiguity concerning the proper cluster affiliation of point x in feature space. Now suppose that the clusters map back primarily to two distinct areas of the image, forming regions R_{C_1} and R_{C_2}. Each of the image points contributing to histogram point x can lie anywhere in the image. Ambiguity could be resolved if it was entirely surrounded in the image by points from either C_1 or C_2. Actually, this ambiguity exists for all histogram points, even those in the middle of a cluster.

The weakness of histogram clustering followed by a mapping of cluster labels back to the image is three-fold:

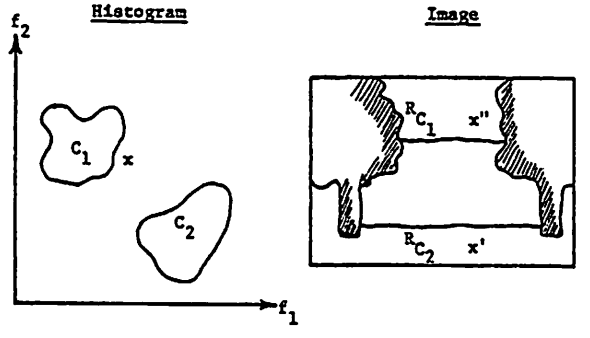


Figure 18. Histograms, Feature Space, and Image Space. The projection of histogram cluster labels back to the image provides only a weak mapping of information between feature space and image space. Consider some point x in the histogram of two features, f_1 and f_2, where its affiliation to cluster C_1 or C_2 is ambiguous. Now assume that R_{C_1} and R_{C_2} are regions produced by the clusters C_1 and C_2, respectively. A pixel contributing to histogram point x may have an image location x' or x'', or in fact lie anywhere else in the image. The problem is more complex since this uncertainty exists even if x is in the cluster core of C_1 or C_2. Decisions regarding the region association of x should be a function of the information in both feature space and image space.

a) formation of clusters in feature space does not take into consideration the spatial distribution of points in the image which formed the clusters;
b) the extent of a cluster in feature space is often ambiguous (e.g., the valley between clusters might be a plateau), and decisions concerning cluster definition are prone to error; and
c) the mapping of a single symbolic cluster label back to the image is only a gross representation of feature space information, which loses the relationship of each point to the cluster as a whole in feature space.

The result is that only a small portion of the information relating image space (spatial relationships of pixels) and feature space(distribution of feature values in the histogram) have been correlated in these algorithms.

IV.3 Relaxation in Image Space Using Feature Clusters

The general idea of the approach we employ [NAG78] is mentioned in [SCH77]. It will take into account both the global information in feature space and the spatial organization of this data in the image space. Instead of mapping a single cluster label back to each image point, the probability that an image point belongs to each of the clusters will be mapped back to the image. This will be accomplished by extracting a representative center point for each cluster and using the relative distance of the feature values of the pixel to these clusters in feature space to determine the probability for each cluster label. The effect is to map most of the information in feature space back into the image where spatial information can be employed. A relaxation labelling process is now rather natural since the probability that an image point belongs to each of N clusters is available. Similar labels will support each other, while different labels will compete over local neighborhoods in the image. As we shall show, the relative cluster distances will be used to determine the compatibility coefficients.

Note that each of the three weaknesses in the process of mapping histogram clustering labels back to the image have either been entirely circumvented or else reduced. The extent of the cluster does not have to be determined, and much of the information in feature space is mapped to the image. Cluster centers still have to be determined and we will say more about this shortly.

IV.4 Assigning Initial Probabilities of Cluster Labels to Image Points

The relaxation labelling process described earlier assumes that given a set of N possible labels, $\lambda_1,\ldots,\lambda_N$, each point in the image has associated probabilities $p(\lambda_1),\ldots,p(\lambda_N)$ that the labels are correct. In the current formulation, the labels will be the cluster identifications and the probabilities reflect the confidence that the image point is a member of that cluster. Thus, the probabilities of the labels for some image

point should be a function of the distance of its position X from each cluster center in feature space. Figure 19 illustrates the situation in the two-dimensional case. Our choice among several possibilities for computing the initial probabilities is

$$p_X(\lambda_i) = \frac{1/d_i}{\sum_{i=1}^{N} 1/d_i} .$$

This choice has the property that the probability is a monotonically decreasing function of the Euclidean distance of the point X from the i^{th} cluster center. The denominator represents a normalization to a true probability.

It will be useful to keep the probabilities of all labels non-zero. Once a label has probability zero it will remain there during relaxation because the updating of probabilities involves a multiplicative function (refer to Section III.5). Therefore, points with $d_i = 0$ (i.e., which are zero distance from the i^{th} cluster center) are treated as a special case; for these points, the i^{th} label will have probability approaching one while other labels are assigned small (but non-zero) values so that they sum to one; thus, all labels will have non-zero probabilities. This will allow the probabilities of other labels to grow if the context so demands, even for image points associated with a cluster center.

An alternative choice for the initial probability computation is

$$p_X(\lambda_i) = \frac{1-d_i/D}{\sum(1-\frac{d_i}{D})} = \frac{D-d_i}{\sum(D-d_i)} \quad \text{where } D = \sum d_i$$

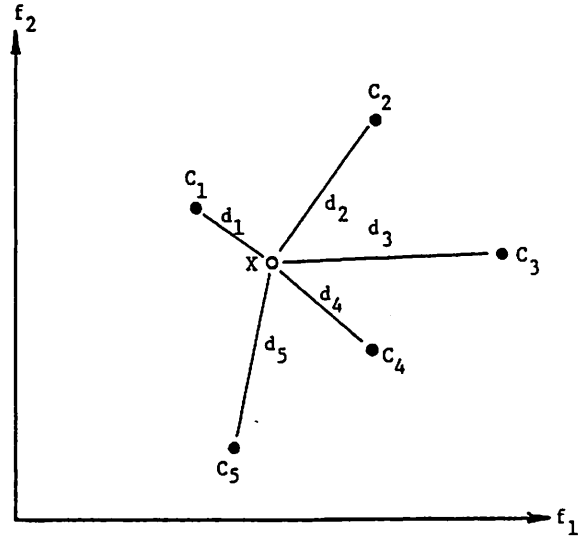


Figure 19. Initial Probabilities of Cluster Labels. The initial probability that histogram point X belongs to cluster C_i will be a function of distance d_i relative to the set of d_j, $j \neq i$.

which de-emphasizes the strength of close clusters whose contribution grew nonlinearly ($\frac{1}{d_i}$) in the first case. The results cited below were obtained using the first probability function.

IV.5 Identification of Cluster Centers in Feature Space

The initial labelling process depends only upon the identification of a single representative prototype point for the center of each cluster in the histogram. It will not be sensitive to relatively small misplacements of the center unless cluster centers are very close. The limits of each cluster no longer have to be determined -- the probability of belonging to a cluster will automatically decrease with distance from the cluster center. A number of techniques can be used to find these centers, but several potential problems must be kept in mind. Since the goal is to eventually obtain regions which are in close correspondence to the objects (or parts of objects) appearing in the scene, the algorithm should minimize the chances of arbitrarily splitting regions due to misidentification of cluster centers. This can occur in two ways: if a cluster is missed or if a cluster is mistaken to be two clusters.

If the clustering algorithm misses an obvious cluster in feature space (and consequently no label for this cluster is defined), the image points comprising this cluster will gravitate towards the clusters which are nearest in feature space. If there is only one, then the net effect will be to absorb the missing cluster into one which has been labelled. This type of error is not serious since recursive application of the region formation process will eventually recover it. On the other hand, if the cluster which is missed happens to lie between two or more clusters, then some of the feature points of the missing cluster may lie closer to one of the identified clusters, while others may be near a different cluster. This is a more difficult error from which to recover. Regions in image space corresponding to the missing cluster could be split and absorbed into different regions, if the regions happen to be associated with the clusters competing for the affiliation of the points in the missing cluster. It is much more difficult to recover from this kind of splitting since local evidence of similarity no longer exists -- the characteristics of the split region can be swamped by each of the regions which absorbed the pieces.

Arbitrary splitting also occurs if a single cluster is identified as two distinct clusters. The result is split regions which also could lead to the problems described above. However, in some cases two adjacent regions could be merged afterwards based upon similarity of their region characteristics.

Let us briefly review several alternatives for the extraction of cluster center representatives. We wish to point out that a variety of clustering algorithms appear in the pattern recognition literature. In pattern recognition applications, clustering algorithms are often applied only once to produce a characterization of the underlying data; in the application discussed here, clustering

is one of many steps in region formation and must be repeated many times during the course of segmenting an image. In this case computational cost is an important factor in the selection of a clustering method.

We will examine this problem from a different viewpoint. The problem of extracting these prototype points is not nearly as difficult for one-dimensional histograms as it is in higher dimensional spaces. In the following discussion, we will consider only one- and two-dimensional histograms; some of the approaches generalize to the n-dimensional case and others do not.

Ohlander [OHL75] has defined a set of rules for cluster detection based on analysis of local peaks and valleys, and their relative distances in one-dimensional histograms. In two dimensions the problem is more difficult because it appears to involve a search in two-space for the worst-case valley between two clusters. If the minimum value on each possible path between clusters represents the degree to which that path is considered to be a valley, then the limiting valley is that path which maximizes across all paths the minimum value on the path. This implies that an examination of all connected paths between the clusters is necessary -- a computationally expensive process which is even worse in higher dimensions.

Another approach is to use a conservative clustering algorithm in an attempt to define cluster cores [HAN75,NAG77]. The two-dimensional histogram is treated as a pseudo-image in the processing cone; it is two-dimensionally averaged by reducing spatial resolution (reduction), and then weak values are thresholded. The effect is to spatially collapse relatively high values of the histogram which are in close spatial proximity into a connected cluster region, while deleting the valleys. A region growing process is then used to label the cluster cores in this reduced resolution histogram. This process is reasonably effective, although the criteria by which the threshold is determined as a function of the reduced values must be carefully studied for reliability. One mechanism that has been used to compute the threshold involves an examination of a histogram of the histogram (i.e., the distribution of histogram values).

An iterative peak enhancement process has been described by Rosenfeld [ROS77b]. On every iteration, each histogram bucket is compared pairwise to each bucket over some predefined neighborhood. The central bucket is increased or decreased as a function of the values in the neighborhood; the amount is directly proportional to the difference in bucket values and inversely proportional to their distance apart. This algorithm can be applied in parallel to all buckets, causing clusters to dynamically organize themselves. It appears quite appealing in that thresholds are not necessary, but it is sensitive (hopefully weakly) to the choice of neighborhood size.

In the following discussion, we assume that a set of N prototype points $X_1,\ldots,X_N$, representing cluster centers in feature space, have been extracted. For the sample results that follow, they have been set by hand, but it should be possible to extract most clusters automatically. This problem is under continuing examination.

IV.6 The Compatibility Coefficients and Updating Probabilities

The compatibility coefficient between each pair of labels defines whether labels of neighboring pixels support each other or compete with each other. The coefficient is positive for identical labels and negative for differing labels. The simplest choice is to have

$r_{ij}(\lambda,\lambda') = 1$ if $\lambda = \lambda'$

$r_{ij}(\lambda,\lambda') = -1$ if $\lambda \neq \lambda'$

Notice that the linear summation across labels (refer to Section III.5) implies that the updating contribution from pixel j to $\Delta p_i(\lambda_K)$ will be zero if the probability of λ_K at location j is equal to .5 and will be negative if the probability is less than .5. However, even if all labels have total contributions which are negative, the probability of that label whose Δp_i is least negative will increase, relative to the other labels.

This simple specification of compatibility coefficients works reasonably well, but it can be improved by introducing relative weights on the coefficients which reflect the confidence that the two clusters really are distinct in feature space. This effect is incorporated for labels λ and λ' simply by scaling its negative contribution by the ratio of the distance between clusters λ and λ' to the maximum distance between any pair of clusters. Let $d_{MAX} = \underset{\lambda,\lambda'}{MAX}[d_{\lambda\lambda'}]$; then

$$r_{ij}(\lambda,\lambda') = -\frac{d_{\lambda\lambda'}}{d_{MAX}} \quad \text{for } \lambda \neq \lambda'.$$

This slows down the changes in label probabilities induced by the relaxation process in ambiguous cases where clusters are close together, and speeds up the change (relatively) in clear cases where clusters are far apart. Note that the most distant pair of clusters will have an $r_{ij} = -1$.

There is one additional problem in the definition of the neighborhood of a region. If an

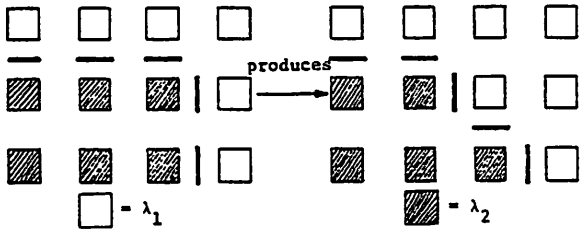


Figure 20. An 8-adjacency Neighborhood Causes Problems at Corners. As the label probabilities converge, the label λ_2 of the corner pixel will have competition from the high probability labels λ_1 at five neighboring pixels, and support from high probability labels λ_2 at only three neighboring pixels. As the neighborhood of labels converge, the corner pixel will switch affiliation from λ_2 to λ_1. The use of a 4-adjacency neighborhood removes this difficulty.

8-neighborhood is employed, right angle corners often cannot survive as probabilities converge to one. Figure 20 shows a pixel with label λ_2 at the corner of a region. In its 8-neighborhood there are only three similar labels of λ_2 and five dissimilar labels. This causes the central pixel at the corner to change affiliation from λ_2 to λ_1 which then produces a stable situation. Use of a 4-neighborhood removes this difficulty, and any particular diagonal element still will have an influence upon the central pixel indirectly via two intermediate neighbors.

It should also be pointed out that this relaxation algorithm forces all pixels to eventually become associated with a cluster, since no null label was defined. It might be argued that those pixels equidistant from two or more clusters, or far removed from any cluster, are ambiguous. It might be advantageous to include a null label in those cases to "catch" the ambiguous pixels, and examine null label regions after convergence on a more global basis. This has not been explored yet. Finally note that patterns of labels could be used in the region relaxation process, just as in the edge relaxation process, if patterns of labels (e.g., corners) have meaning.

IV.7 Results of Relaxation on Cluster Labels

The intensity (I) of the suburban house scene shown in Figure 21a is used to demonstrate this algorithm. The RGB color data is transformed to produce the red-cyan ($\overline{R}$) feature which is shown in Figure 21b. The one-dimensional histograms of I and $\overline{R}$ are shown in Figure 21c and d. These two features produce the two-dimensional histogram shown in Figure 21e; the six cluster centers that were selected are circled, of which all but two are quite obvious; the remaining two should be extractable by improved clustering algorithms.

Figure 22 shows the results obtained by projecting this information back to the image. The images in the first six columns display the probability of each of the six cluster labels at each pixel, where brightness is proportional to the label probability -- black representing a probability of zero and white a probability of one. The last column is an image formed by selecting the maximum probability cluster label for each pixel; each label is then encoded as a distinct gray level and the image displayed. The row organization is by number of iterations -- the first row being iteration zero (i.e., initial probabilities), followed by results after 1, 3, and 5 iterations.

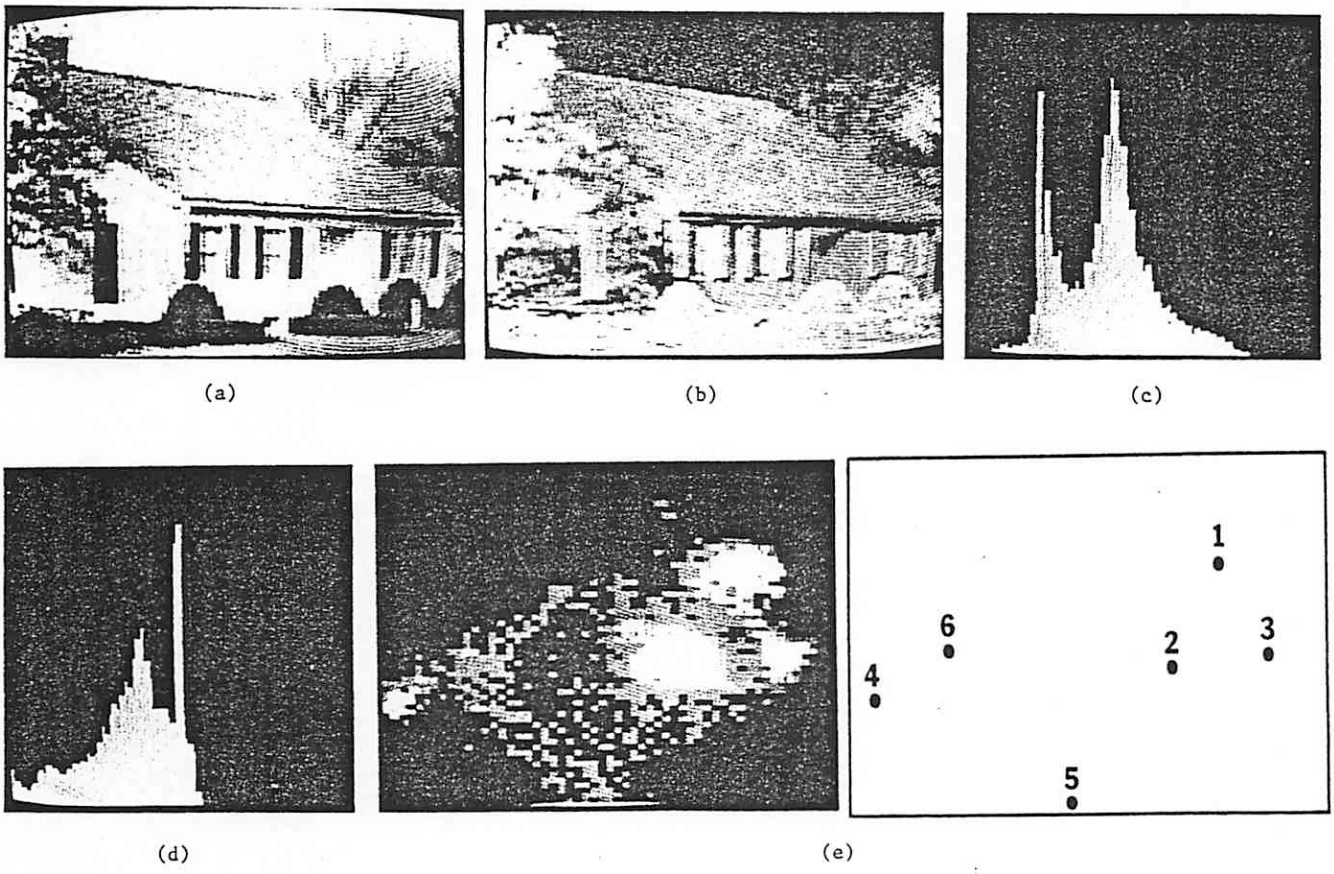


Figure 21. Features used in Region Formation. (a) Intensity image of suburban house scene. (b) The scene in terms of $\overline{R}$, the red-cyan opponent. (c) One-dimensional histogram of I (intensity). (d) One-dimensional histogram of $\overline{R}$ (red-cyan). (e) Two-dimensional histogram of intensity (x-axis) versus $\overline{R}$ (y-axis; cyan is at the top and red is at the bottom); the six cluster centers that were selected are shown in the sketch on the right.

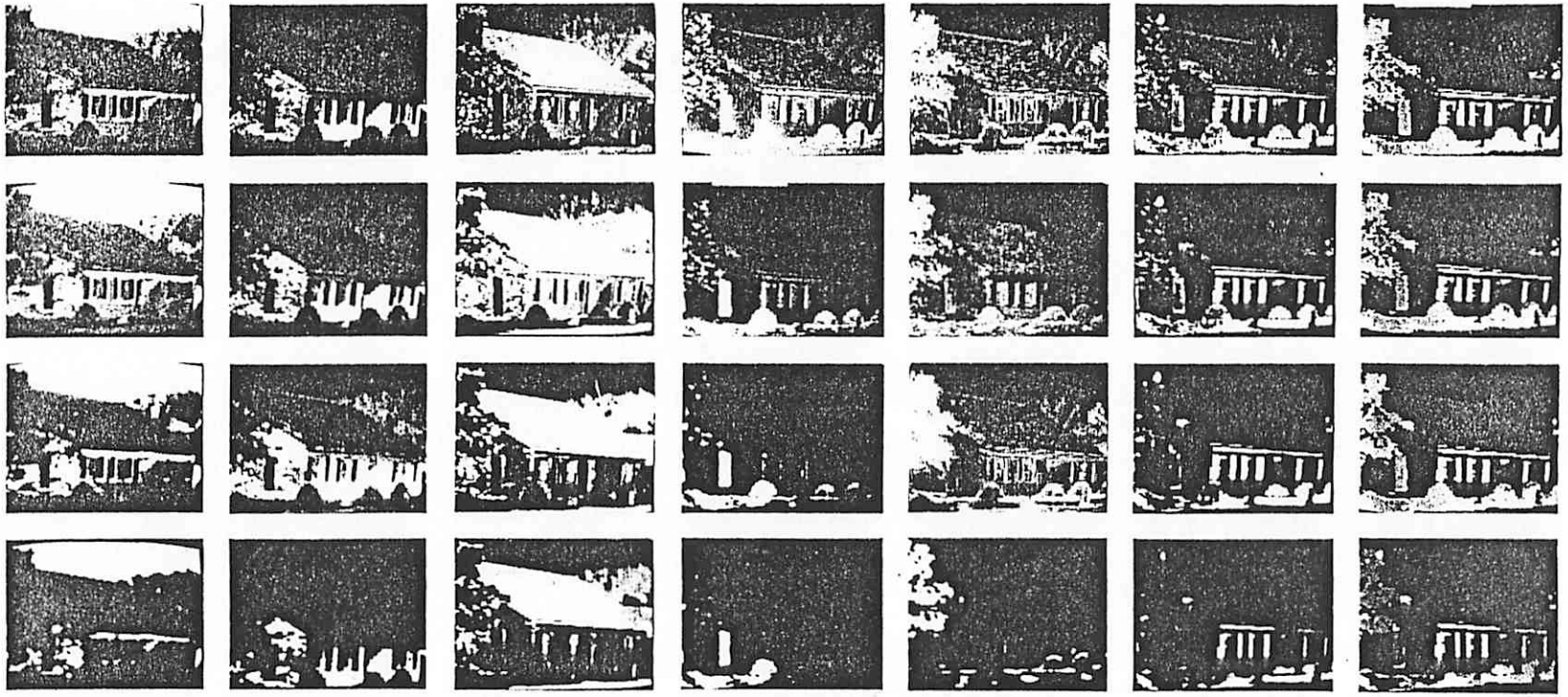


Figure 22. Results of Region Formation. Images in each of the first six columns represent the spatial distribution of the probabilities of labels of a particular cluster. Brightness is proportional to probability with black being 0. Images in the last column are formed by selecting the cluster label with maximum probability for each pixel; six distinct gray levels are used to denote the clusters. The rows represent iterations of the relaxation process.

The relaxation process, making use of both spatial proximity and feature value proximity, effectively extracts many of the globally interesting regions in the image. In the next section, we discuss techniques for further segmenting the image based on these results.

IV.8 Hierarchical Decomposition of the Image

In order to obtain a useful decomposition of the image in terms of regions, there remain two problems to be overcome:

1) the "correct" level of decomposition is still not known; and
2) histogram clusters can still be "hidden" by larger or stronger clusters due to histogram overlap.

However, we have pointed out that any given scene depicted in an image admits to various levels of description; for example as an outdoor scene; or house, trees, sky, and grass; or windows, doors, roof, leaves, blue sky, clouds, blades of grass, etc. Clearly, each of these descriptions is appropriate given particular goals. In addition the physical distance to objects affects the resolution and thereby the information in the image, as portrayed in Figure 2. It is certainly not the responsibility of a general (non-semantically directed) region analysis to favor one interpretation over the other. Therefore, we are examining a multi-level description of the scene based on region properties.[1] A region at any stage can be related to the parent region derived from a coarser degree of segmentation and to descendant subregions using finer degrees of segmentation. These results could be stored hierarchically [FRE76], where relationships between ancestors and descendants represent descriptive properties of the structure of the visual elements.

Unrestricted recursive decomposition of regions and subregions could lead to serious computational problems when the number of regions in scenes with texture increase exponentially as finer and finer detail is extracted. This is clearly not a desirable effect of the algorithm. When a region breaks up into many smaller regions, possibly representing textural elements, the parent region ought to be given appropriate

[1]This research, currently in progress, is being conducted by Paul Nagin of the COINS Department at the University of Massachusetts.

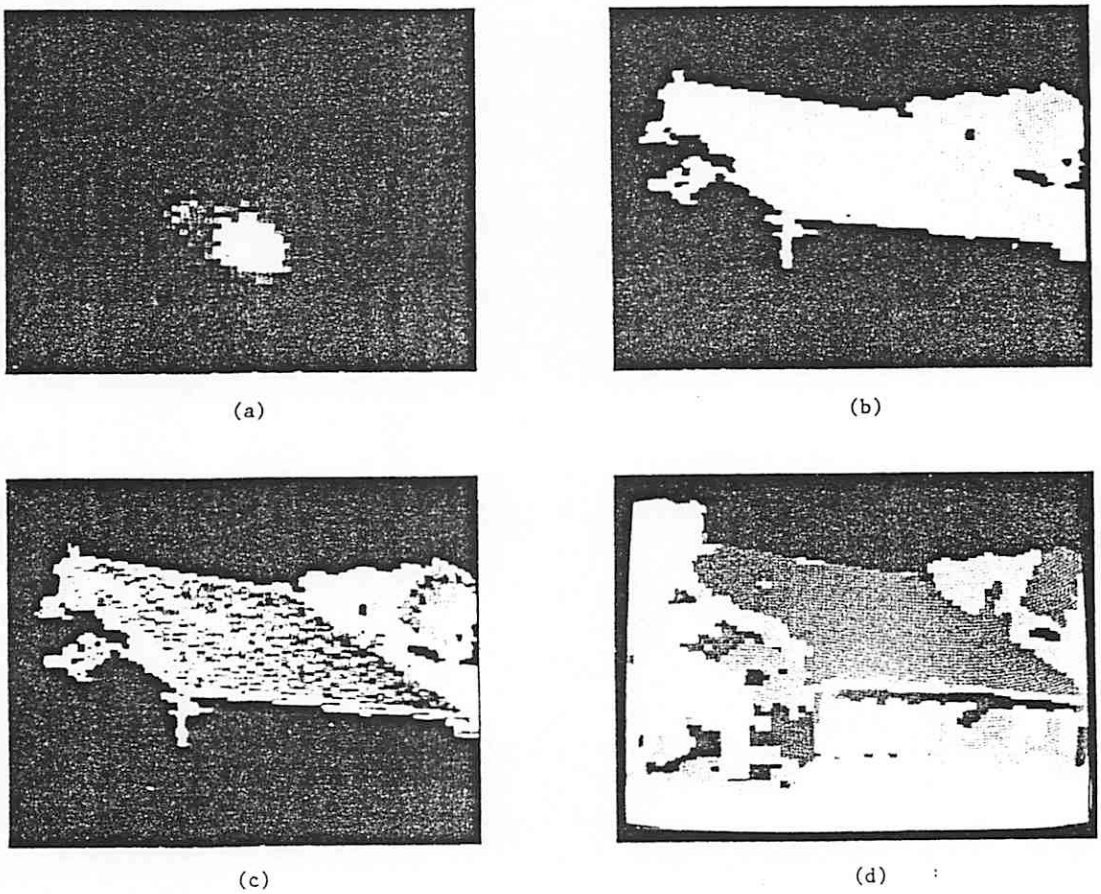


Figure 23. Recursive Segmentation of the Roof Area. (a) 2D histogram of overmerged roof region in Figure 22 forms a major and a minor cluster. (b) Initial probabilities for major cluster. (c) Initial probabilities for minor cluster. (d) Final segmentation with recursive analysis of roof area incorporated into results of Figure 22.

descriptors of the decomposition without saving each specific subregion; only information on the spatial distribution of the distinct types of subregions (e.g., red and yellow polka dots on clothing) need be saved. On the other hand a textured tree region might have descendant arcs to large subregions representing macro-texture elements, such as large dark shadow areas or large leafy branch clumps; here each large subregion would be explicitly represented.

The process by which the hierarchical decomposition would be obtained also alleviates (but does not remove) the hidden cluster problem if we base this process on a multi-stage recursive histogram analysis [OHL75]. The purpose of the first stages of this analysis can be thought of as "planning" [KEL71,HAN74,PRI77], where only gross structure is extracted by reducing the amount of detail in a scene to a bare minimum. Then, each of the grossly overmerged regions can be carefully refined [NAG77]. Thus, refined regions at one level become plans at the next level, with the histogram overlap confined to the plan regions instead of the whole image.

The idea of refinement is demonstrated in Figure 23 where the roof was partitioned by a recursive pass on that portion of the image. The house roof, garage roof, and the tree with bare branches all have very similar features. When the 2D histogram is confined to only the overmerged roof region, the subtle visual differences in these areas appear as a major cluster with a nearby minor cluster. An image of a farmhouse and histograms of two features are shown in Figure 24. It is partitioned in Figure 25 using three clusters, and produces an effective plan for further decomposition in the manner described.

IV.9 Guidance from High-Level Interpretation

This region formation approach delegates to the high-level system the responsibility for sorting out the correct level of description in the hierarchical segmentation by fitting interpretations to different levels, or by extracting useful levels depending on size and properties of regions. The final output of the low-level region analysis system is a hierarchical segmentation providing increasing levels of detail in the scene. In

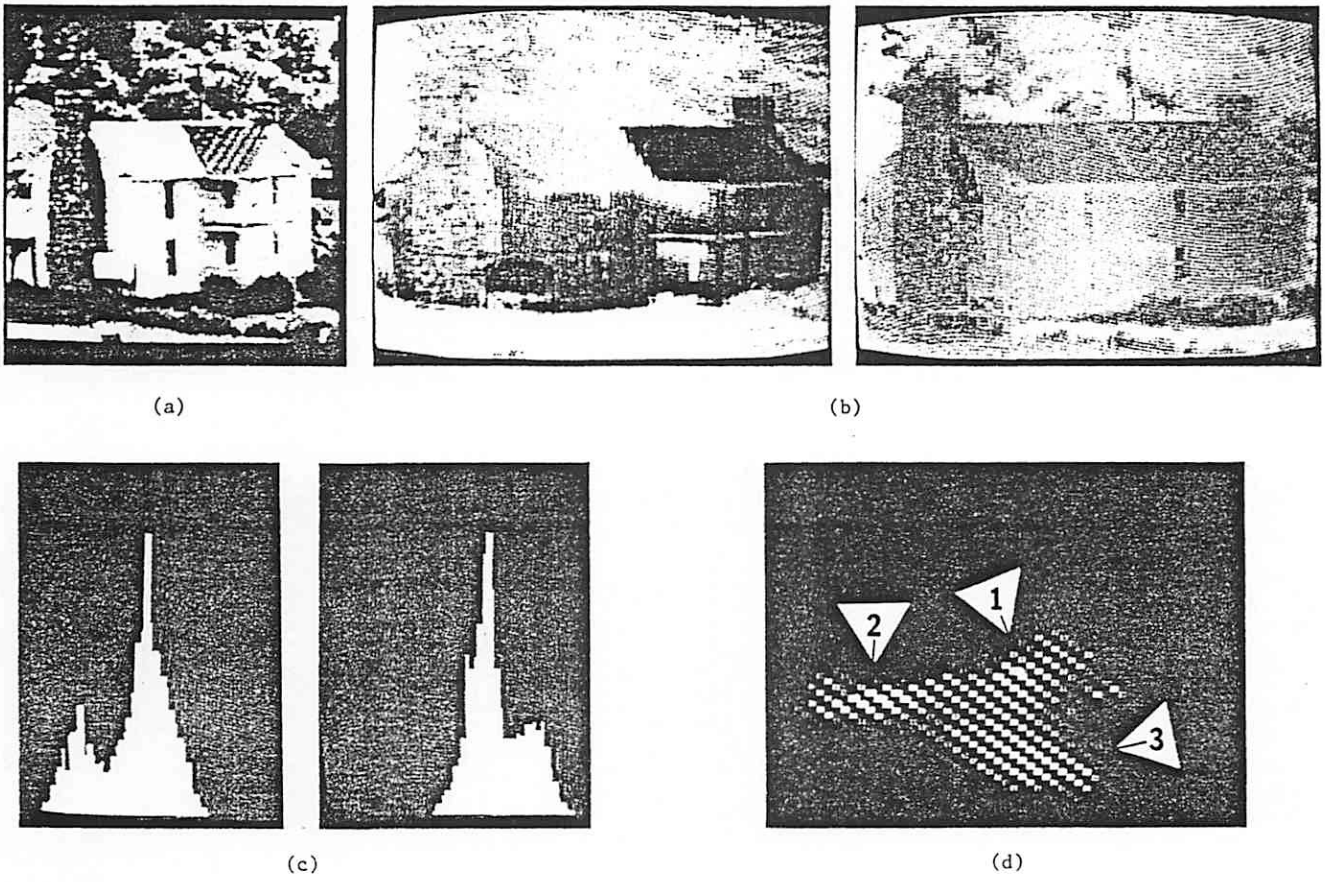


Figure 24. Features used in Region Formation. (a) Intensity image of farmhouse scene. (b) The scene in terms of $\overline{R}$, the red-cyan opponent (shown on left) and $\overline{G}$, the green-magenta opponent. (c) The one-dimensional histograms of features used, $\overline{R}$ (left) and $\overline{G}$ (right). (d) Two-dimensional histogram of $\overline{R}$ (x-axis; cyan is to the left and red towards the right) and $\overline{G}$ (y-axis; green is at the bottom and magenta at the top); the three clusters that were selected are shown. Note that the orderly appearance of the histogram entries is due to the feature dependencies noted in Section IV.1

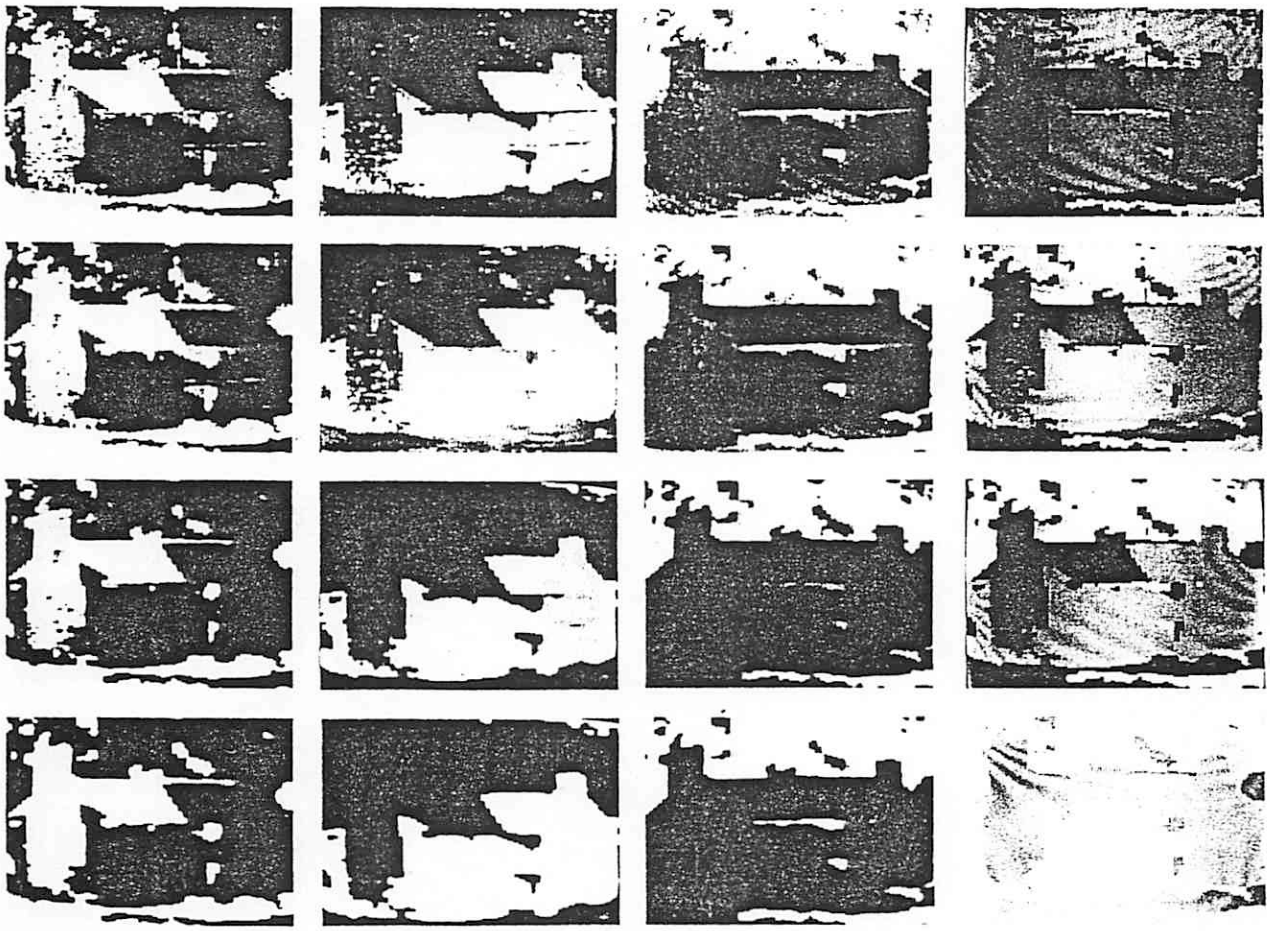


Figure 25. Results of Region Formation. Images in each of the first three columns represent the spatial distribution of the probabilties of labels of a particular cluster. Brightness is proportional to probability with black being 0. Images in the last column are formed by selecting the cluster label with maximum probability for each pixel; three distinct gray levels are used to denote the clusters. The rows represent iterations of the relaxation process.

addition, the refinement process could be guided by feedback from object verification strategies so that some portions of the hierarchical plan representation would be refined to a greater level of detail, using features carefully selected for recognition and discrimination. Here, a comparison of region attributes at any level of the hierarchy with stored object attributes can determine good hypotheses for region identities. Ambiguity in hypotheses can direct feature extraction for discrimination between hypotheses, while lack of a good match could lead to further decomposition and matching. The linkage between the region segmentation process and the interpretation system is still in the design stage.[1]

V. Merging Multiple Representations

Since regions and closed boundaries implicitly define each other, we have selected the edge/boundary representation as the common form in which to compare results from the two algorithms. The set of regions that have been produced imply boundary segments in terms of interpixel edges. The goal is to integrate the edge and region information to produce a segmentation of the image which is more reliable than either.[2]

It is not surprising that the results obtained from the two analyses do not agree in some portions of the image. Consider the set of tree trunk images

[1]This work is currently under development by Paul Nagin and Tom Williams of the COINS Department, University of Massachusetts.

[2]This research, currently in progress (and continuation of that reported in Section III) is being conducted by Ralf Kohler of the COINS Department at the University of Massachusetts.

in Figure 2, and the disparate micro/macro views of the data which would be afforded the two analyses. In addition the results we have presented in this paper have been obtained using different features, monochromatic intensity for the boundaries and color features for the region analysis.

The simplest correlation of the two segmentations is to perform a direct intersection of the results. The edges which survive are those which have been produced in exactly the same location by both the region and edge analyses. However, it is to be expected that the placement of a boundary will often be ambiguous, particularly where there are wide gradients. In such places the edge analysis forms a decision of good line continutation as a function of the gradient model used to determine edge placement during gradient collection and the interaction of local contexts during relaxation. In region analysis, wide gradients can be viewed as a slow transition from one region to another, appearing in the histogram as a trail of points connecting one cluster to another. Where the regions terminate (i.e., where the boundary between two regions is located) depends upon the spatial relationships and feature cluster relationships of the pixels in this area to the regions they lie between.

To allow for small differences in the results, a "k-intersection" is defined, in which the edges within k pixels of each other appear in the intersection. The results in Figure 26 show the intersections for k = 0,1, and 2. In this figure, exact (i.e., 0-intersection) matches are shown by the brightest boundaries, matches within one pixel (1-intersection) are medium bright, while the lowest intensity boundaries are 2-intersection matches.

Rather than using such an arbitrary intersection of the results, the spatial extent of the gradient (used to collect edges) can provide a local view of the alternative placements of the boundary. For large gradient widths, the acceptable disparity between region-induced boundaries and those from the edge analysis can be increased. Investigation of this dynamic intersection process, and the problem of where to place the resultant edge, is currently under way. A sophisticated method of integrating segmentation results ought to use the width of the gradient in the boundary analysis, the confidences associated with elements in each segmentation, an understanding of the weakness of the algorithms in areas of texture, and the incorporation of gestalt laws of line continuation and completion. A variety of strategies are being examined here, but are still in an early stage of development.

In the remainder of this section we outline a measure of confidence for boundary segments that has two purposes. First, it will be used in resolving conflicts between the representations. Second, the merged results, symbolically labelled and with associated descriptors, form the input to the interpretation processes (see Section VII and the companion paper) and it will be quite useful to have a measure of reliability of the various portions of the segmentation. These confidences could then be used to determine the confidence of less primitive constructs (for example, primitive two-dimensional shapes; see next section).

The confidence in the segment S will be based on information in the neighborhoods T_1 and T_2 to either side of the segment. These neighborhoods will be defined differently in order to reflect the data used by each algorithm. An interesting measure of confidence involves a test of the hypothesis H_1 that the points in T_1 and T_2 belong to different regions, against hypothesis H_0 that points in T_1 and T_2 belong to the same region; this measure was originally developed to guide the boundary formation process itself [YAK76].

This statistical test is defined as follows. Let $T_0 = T_1 \cup T_2$. Under the hypothesis H_0, the set of pixels T_0 is a single region modelled by the normal distribution $N(\mu_0,\sigma_0)$; under the hypothesis H_1 the two sets of pixels T_1 and T_2 are modelled by distributions $N(\mu_1,\sigma_1)$ and $N(\mu_2,\sigma_2)$, respectively. A maximum-likelihood analysis leads to the measure of confidence in H_1 as

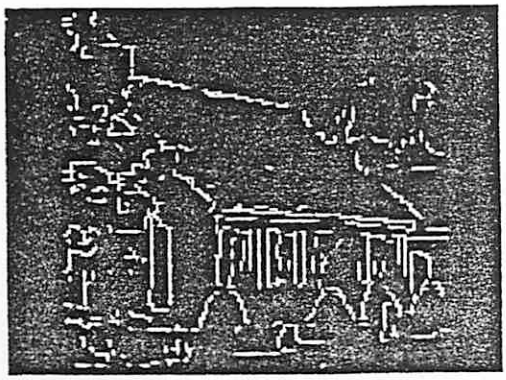


Figure 26. Results of Merging Boundary and Region Data. Exact boundary matches (k=0 intersection) are shown by the brightest boundaries. Matches within one pixel (k=1 intersection) are medium bright, while matches within two pixels (k=2 intersection) are of lowest intensity (note that these latter matches may not be visible in the figure as reproduced here).

$$\frac{(\sigma_0^2)^{n_0}}{(\sigma_1^2)^{n_1}(\sigma_2^2)^{n_2}}$$

where n_i is the number of pixels in T_i, and $n_0 = n_1 + n_2$. The actual test we utilize is an extension of this formula to account for (linear) spatial gradients in the data as opposed to the assumption that the normal distributions are independent of their location in the image; more details will appear in [PRA78].

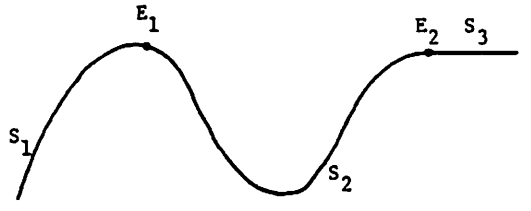
The only difference in the computation of confidences for the edge/boundary-produced segments and the region-produced segments is the definition of the neighborhoods of T_1 and T_2. In the case of the edge analysis, the data that produced the segment was relatively local; consequently T_i will be defined to include pixels within a distance k (for our application k = 2) to one side of the segment. In the region analysis the data was more global and therefore T_i will be defined to include all the pixels in a region to one side of the boundary. Thus, the confidence of a segment reflects the local or global data which was involved in the computation of that segment.

VI. Curve Fitting and 2D Shape

The results obtained from the segmentation processes and the merging operation will be symbolically labelled regions and boundaries (to be discussed in Section VII), but the representation of the boundaries (either regions or edges) is still in a form very close to the original data (e.g., x-y coordinates). One of the justifications for the horizontal/vertical inter-pixel edge representation was that decisions concerning global attributes of boundaries (for example slope) should be delayed until information over larger areas of the image has been organized. It is at this point that we wish to attach to each region and boundary segment some of the appropriate symbolic attributes of shape which will eventually allow access to objects stored in the knowledge base [YOR78a]. These attributes are related to the 2D properties of the boundaries; extraction of other properties, for example, those relating to the transition from 2D to 3D space, are delayed until even more information is examined and utilized ([HAN78], this volume).

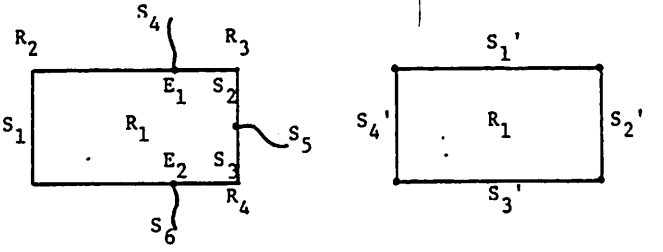
Each boundary segment must be fit with a smooth curve, the type of curve then becoming one of the attributes of that segment. In addition, each region (described in terms of the segments forming the full perimeter of the region) will be examined for good fits by conics [AGI72,SHI78], triangles, rectangles, and polygons in general [FEN75,PAV75].

Prior to performing the fitting analyses, however, there are several difficulties to be overcome. The first is shown in Figure 27a. Let us suppose that the smooth curve shown there has been broken into three segments S_1, S_2, and S_3. The best third degree fit for each segment will not necessarily join smoothly at their endpoints. As a



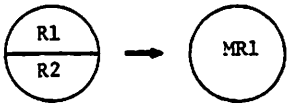
(a)

Segments individually fit with smooth curves must be joined together smoothly. This can be achieved by using splines.



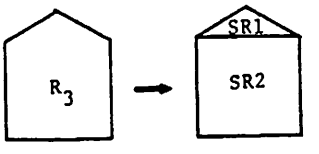
(b)

The segments bounding a region must be restructured by choosing new vertices for further processing of shape.



(c)

Regions sometimes must be merged before a primitive shape can be associated with regions. This problem can be produced by shadows (which may also provide cues for directing this merging.



(d)

A region may have to be split in order to fit simple primitive shapes.

Figure 27. Additional Considerations in Shape Fitting.

result, the fits of segments S_1 and S_2 may not have the same tangent at E_1, and therefore arbitrary discontinuities (as opposed to the inherent discontinuities at the corners of the rectangle in Figure 27b) are introduced into the representation. This problem is eliminated by using cubic splines [AHL67,GOR74], which provides a fit using piecewise third degree polynomials such that the functional approximations come together smoothly at the point of their meeting. A number of difficulties arise when using conic fits [AGI72] and it is possible that the spline fits will replace them; examination of plausible solutions to these problems is still in progress.[1]

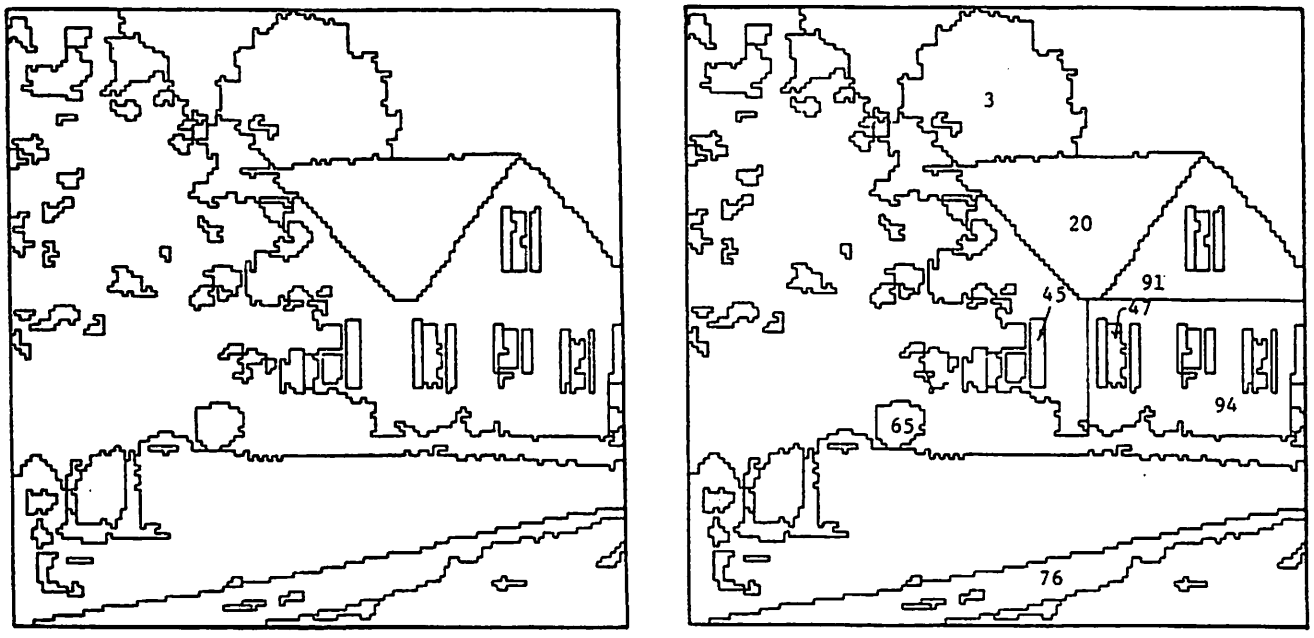
The second problem requires resegmentation of the boundary segments. The initial segmentation was a function of the way the 3D surfaces of a scene happened to project on the image plane from a particular view. The boundary segments that have been extracted lie between at most two regions and have been broken at vertices where three segments come together, or where segments terminate. If segment S_1 (terminated by vertices E_1 and E_2) of Figure 27b were to be fit as shown, then some second degree curve would probably give the least root mean square (RMS) error. More importantly the colinear sections of S_1 and S_2 at the top of R_1 would not be joined together into one side of the rectangle R_1. Figure 27b shows the resegmentation that is necessary before curve fitting in order to eventually arrive at a labelling of R_1 as rectangle. An analysis of points of high relative curvature within a segment and similarities between adjacent segments can be employed to find the points on the boundary of a region which allow more meaningful fits of curves. There are obvious problems that must be handled; for example the best points for resegmentation of the boundary of two adjacent regions will not necessarily coincide on their common boundary.

Finally, let us consider a third problem. Before the fit of regions should be attempted, adjacent regions may need to be split or merged as shown in Figure 27c-d. These kinds of problems are not necessarily caused by incorrect segmentation and cannot be avoided; for example, shadows projected on surfaces, or silhouettes, can lead to cases where this phenomena occurs. We are examining the geometric cues for automatically splitting regions, but this has been carried out by hand in Figure 28b.

The results shown in this section do not make use of our most recent segmentation algorithms as presented in the previous sections. The data used to illustrate the 2D shape processing is the segmentation, shown in Figure 28a, based solely on an earlier region analysis[2] [NAG77] which did not incorporate the link between histogram space and image space as discussed in Section IV. As we described in the last paragraph, the house body has

[1]This work is being conducted by Bryant York of the COINS Department, University of Massachusetts.

[2]One of the problems is that segmentation is carried out on our PDP-15 laboratory machine while the remainder of the processing is done on the University Computing Center's CDC Cyber 74. Data must be converted and transported from one large system to another. This delay will be eliminated by a new laboratory machine soon to be installed; both systems (segmentation and interpretation) will be integrated on this machine.



(a) The segmented image.

(b) The segmented image with two straight lines added to split the major house region. The regions referred to in the discussion are numbered.

Figure 28. Segmented Image for the 2D Shape Analysis.

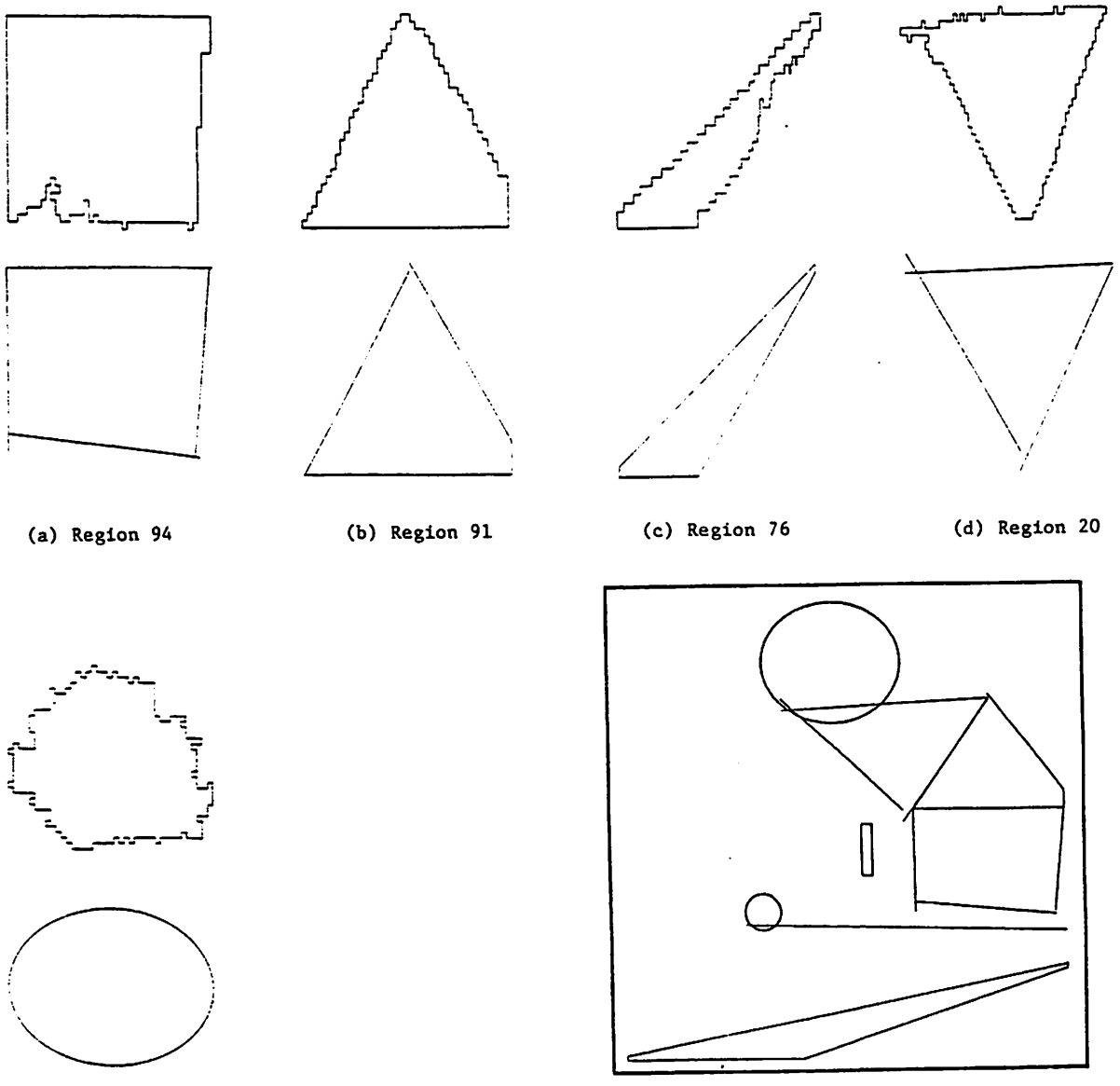


Figure 29. Shape Fits for Selected Regions and Segments.

been split by hand by inserting two straight lines into Figure 28a to produce the image in Figure 28b. A subset of these regions have been fit and are shown in Figure 29; the composite image obtained from these fits is also shown there. Once the regions have been resegmented and the fits for line segments have been determined, the set of segments bounding a region can be examined for fits by primitive shapes, such as squares, rectangles, quadrilaterals, triangles, ellipses, etc. Stored world knowledge contains the description of surfaces, volumes, etc. in terms of these shape labels, and object hypotheses can be made on the basis of the labels and the confidences associated with them. Methods for computing these confidences from the root mean square error of the individual segment fits, as well as geometric constraints on the relationships between segments, is still in an early stage of development and will not be reported here (however, see [HAN78] for further discussion). Table I summarizes the results obtained from straight line fits to several of the regions shown in Figure 28b.

It is desirable to allow a good straight line fit to be maintained despite the possibility of a marginally better fit by second or third degree curves. This acknowledges the semantic importance of straight lines and the probable errors that creep in during segmentation due to discretization.

Region	RMS Errors of fits for segments	Average RMS error of boundary
20	1.53, .93, .48	.98
45	0, 0, 0	0
47	0, 0, 7.59, .43	2.01
76	0, .48, 0, 1.15	.41
91	.723, 1.50, 0	.74
94	0, 4.59, 1.49, 0	1.52

Table I. Straight line fits to selected segments of Figure 28b.

Similarly, second order curves that have low mean square error will be preferred to marginally better third order fits. Of course the use of such precedence leads to ambiguity when the RMS errors of the fits do not provide a clear decision; in such cases both fits should be carried forward.

VII. The RSV Graph: A Symbolic Representation for Segmented Images

We have already motivated the reasons for a transformation of the spatial array of numeric data into a symbolic representation. This representation can be organized via some of the natural spatial relationships which exist between the segmentation elements which are extracted; regions can be defined by the boundary segments which delineate them, and segments can be terminated by vertices (which also have visual properties).

The representation chosen for a segmented 2D image is a layered graph, known as the RSV[1] graph, containing three planes in which the symbolic labels for regions, boundary segments, and vertices appear as nodes; relevant attributes are attached to each node. The RSV graph captures the visual spatial syntax of the image in the arcs between entities, allowing these relationships to be easily accessed and manipulated. Interpretation processes will need access to, for example, the segments which bound a region and the regions which are adjacent to each other. At least these properties should be directly accessible in the RSV graph.

The directed layered graph provides a convenient and flexible representation for the image data -- it provides a repository for features extracted from collections of image points, since each node on a plane (R, S, or V) represents specific instances of that class of entities in the image. The arcs between the nodes (both inter- and intra-planar) capture important two-dimensional, spatial relationships. These relationships (between nodes at one level) appear as a node at other levels in the representation. For example, the fundamental relationship of region adjacency is implicitly available as a segment node; it is represented by an arc from each region node to their common line segment node. Thus, a region is defined by the set of line segments which form its boundary.

Regions which are adjacent to each other in the segmentation are a function of the original scene and the way three-dimensional surfaces project onto the two-dimensional image. The visual properties (e.g., contrast) of the boundary of a region can be expected to vary in arbitrary ways as the boundary is traversed. However, the properties of segments which lie between a pair of regions can be expected to remain far more invariant. In RSV, a line segment is defined by a pair of adjacent regions, unless it is a nonbounding segment contained in a single region: this is a highly desirable property of the representation.

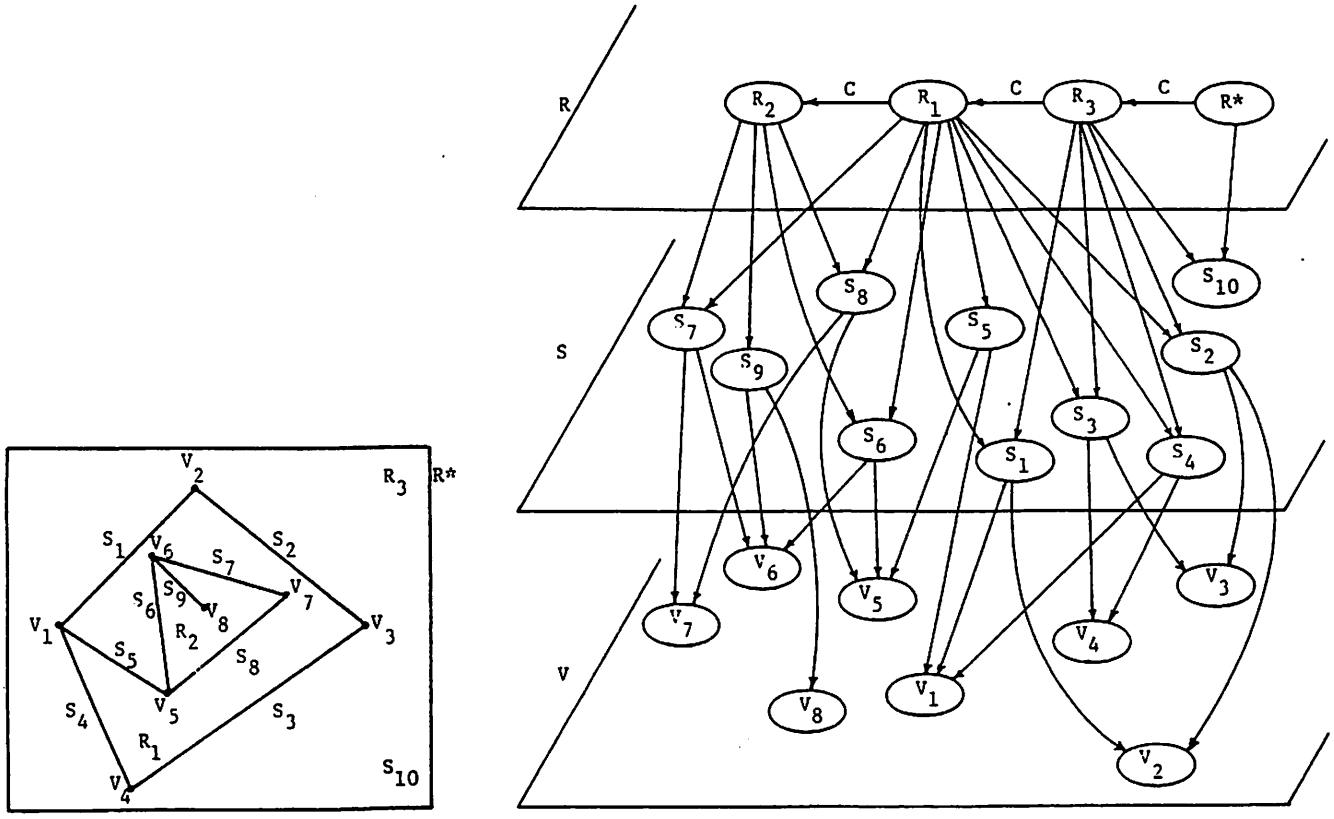
A line segment is anchored in two space by the position of its terminating vertices.[2] A vertex of degree 0 is an isolated point, a vertex of degree 1 is an endpoint of a single segment (no continuation), and a vertex of degree 2 is a junction of two distinct segments. In general, a vertex of degree n is the junction of n distinct line segments. Vertices of degree 1 or degree ≥ 3, by definition, cause termination of a segment. The meeting of line segments is represented by arcs to their common vertex node. If regions must later be split or joined as object recognition proceeds, this representation affords the flexibility for redirecting a few pointers to update the low-level visual data.

Figure 30 is a simple illustrative example. The R plane projects down upon the S plane which in turn projects down upon the V plane. In particular, regions are enclosed by segments (and can, in turn, enclose non-bounding line segments, such as S_9). Note that R* is a special region which includes everything outside the picture and therefore points to the line segment(s) on the boundary of the image. In this example we have further subdivided a boundary segment into segments which are straight when this property is applicable (except in the case of the picture boundary where they are artificial). Obviously other properties, such as points of high curvature (see Section V), also could be used to subdivide segments. Also note that segments such as S_5 and S_9 which are entirely contained in a region will be called "isolated" segments and are immediately apparent because only a single region node points to them.

Only a limited subset of the possible two-dimensional spatial relationships between regions and lines are being used to form the logical structure of this layered network, namely the adjacency of regions mapping onto segments and connectedness of segments mapping onto vertices. There are many other relationships that can be represented and extracted as explicitly labelled directed arcs between nodes on a plane. For example, containment of region R_2 by region R_1 can

[1] In previous papers RSV was called RSE since the term "endpoints" was used instead of "vertices." Originally only the termination points of boundary segments were employed, but we have generalized this definition as described later in this section.

[2] A line segment which is closed has no vertices; in this case an arbitrary vertex of degree 2 is selected so that the location of the line can be fixed in the image.



(a)

The segmented image with regions (R_I), segments (S_I), and vertices (V_I) labelled. Vertices have also been inserted to mark the ends of straight line segments.

(b)

The layered graph structure represents these results symbolically. Regions are linked by arcs to their boundary segments, which are linked by arcs to the vertices at their ends. Arcs representing other relationships such as region containment (C) can be added. R* is a special region representing everything outside the image; it points to segments on the boundary of the image.

Figure 30. Illustrative Example of RSV Structure.

be represented by a "C" arc from R_1 to R_2. In this way the syntactic graph representation can be enriched by the addition of any further relationships that may be necessary, such as arcs between parallel line segments or endpoints near each other.

The descriptive properties associated with nodes include:

- regions -- location, color, saturation, brightness, texture, size, shape, orientation, centroid, horizontal and vertical extent, etc.
- boundary segments -- location, length, contrast, width of gradient, orientation, shape, etc.; and
- vertices -- location, orientation, type (such as the polyhedral fork, arrow, T), etc.

Given the symbolic RSV graph it is straightforward to define routines to determine:

1) the common boundary segments of two regions,
2) the set of regions adjacent to a given region,
3) the connected boundary around a region,
4) line dominance cues on segments emanating from a vertex ([HAN78a], this volume).

The operation of determining region containment is not at all straightforward in the RSV graph once the spatial array representation is discarded. It is locally unclear whether R_2 contains R_1 which contains R_3, or whether R_3 contains R_1 which contains R_2. One must recursively trace out to the picture boundary and R* to determine the correct case. Consequently, we assume that containment relationships are computed directly from the image and C arcs placed in the R plane. Once C arcs are available, it is easy to extend operation (3) above to determine inner and outer boundaries for regions which contain other regions. More details on these operations appear in [HAN76].

The RSV structure thus provides a transitional representation between the spatially organized image data and the symbolic description of the image in terms of the high-level visual entities of surfaces,

volumes, and objects. It also provides a decomposition between the development of segmentation and feature extraction processes and the development of the high-level interpretation processes. It is a well-defined interface between these two systems, providing a repository and organization for the output of the segmentation system, and the source of input data for the interpretation system.

VIII. Conclusion

The data structures, processes, and algorithms that have been discussed here are a step toward the construction of a general low-level segmentation system which confronts a broad range of problems in vision. They have reached a point where they appear to provide usuable results for interpretation processes. While we do not consider these results as preliminary, neither should they be considered, in any sense, final. There are further extensions and improvement to each of the algorithms currently under study, and feedback from semantic processes seems to have great potential.

The two avenues to initial image segmentation are based on complementary approaches to the extraction of visual information from the image. The boundary formation process responds to local changes in the data while the region formation process is sensitive to global similarities in the data. The edge/boundary analysis provides a representation of local feature discontinuities as a collection of horizontal and vertical edges. The iterative edge aggregation processes then allow semi-global contextual interactions to organize collections of edges into boundary segments through a paradigm of local cooperation and competition. Our approach to region analysis makes use of a gross global analysis of similarity over one or more selected features. Information derived from cluster analysis in feature space is mapped back to the image, followed by spatial adjacency interactions which organize the pixels into regions. Utilizing multiple features and processes based on histogram clustering and relaxation methods, recursive refinement of the decomposition provides a hierarchical description of the scene in terms of regions, each with properties that are invariant (where invariance is computed relative to the particular image).

Both the boundary and region formation algorithms can be extended and work continues in this direction. The relaxation processes which aggregate the edge data ought to be anchored to the original data in terms of the gradient widths which have been extracted and the region characteristics to each side of the gradient; this characteristic has already been explored and appears promising, but requires further investigation. Also, the local neighborhood around an edge can be enlarged so that the patterns for neighborhood edge support have a wider view of the context in which the edge resides. The global region analysis might be improved by the addition of a null label for image points which are ambiguous in feature space, rather than forcing affiliation to a particular cluster. The problems of cluster identification, and the effects of both mistakenly splitting a cluster or missing a cluster must be better understood. Finally, the utility of hierarchical region descriptions must be assessed.

The techniques for merging multiple representations is still at an early stage of development. The topology of the raw information in terms of the gradient widths and expected effects of textural variation on each of the algorithms are being considered. Confidences of boundaries between regions are being supplied by each of the algorithms to direct the merging.

The integration of the relaxation processes for regions and boundaries into a single relaxation process is possible [ZUC77], and this might also be extended to hierarchical relaxation [DAV78] in our processing cone. Segmentations from other sources could be merged to provide information that neither of our current segmentation algorithms have access to. In particular we and some of our colleagues are studying motion cues across a temporal sequence of images, and researchers at other institutions have extracted range data via stereopsis or laser-range devices.

The evaluation of the effectiveness of segmentation processes cannot be divorced from the goals of the semantic processes employing the segmentation results. Thus, the manner and degree of refinement of our low-level system must take into account the quality and effectiveness of the interpretation. With feedback from high-level processing, many problems which are unsolvable during "start-up" of segmentation algorithms yield rich alternative solutions. The evaluation of the segmentation processes in VISIONS must await the evaluation of the VISIONS system as a whole.

Acknowledgements

This research was supported in part by the Office of Naval Research under Contract N00014-75-C-0459. We would like to express our gratitude to the Computer and Information Science Department at the University of Massachusetts for providing a congenial atmosphere in which to conduct this research. The support, constructive criticism, and advice offered by many of colleagues is greatly appreciated. We thank Azriel Rosenfeld for constructive suggestions at various points in the long development of these ideas. The research reported here was made possible by the untiring efforts of Ralf Kohler, Paul Nagin, John Prager, Tom Williams, and Bryant York upon whose research this paper is based. These individuals and those who have contributed to the interpretation system of VISIONS have formed a rare cooperative research group whose dedication, and sometimes personal sacrifice, will always be appreciated. Finally, we wish to thank Ms. Janet Turnbull for her help and patience in producing the various drafts of this manuscript and many others during the past three years.

References

[AGI72] G.J. Agin, "Representation and Description of Curved Objects," Stanford AI Memo 73, 1972.

[AHL67] Ahlberg, Nilson, Walsh, Theory of Splines and Their Application, Academic Press, New York, 1967.

[BAL78] D.H. Ballard, C.M. Brown, and J.A. Feldman, "An Approach to Knowledge-Directed Image

Analysis," in Computer Vision Systems (A. Hanson and E. Riseman, Eds.), Academic Press, New York, 1978.

[BAR78] H.G. Barrow and J.M. Tenenbaum, "Recovering Intrinsic Scene Characteristics from Images," in Computer Vision Systems (A. Hanson and E. Riseman, Eds.), Academic Press, New York, 1978.

[BRI70] C.R. Brice and C.L. Fennema, "Scene Analysis Using Regions," Artificial Intelligence, 1, 205-226, 1970.

[COR70] T.N. Cornsweet, Visual Perception, Academic Press, New York, 1970.

[DAV75] L.S. Davis, "A Survey of Edge Detection Techniques," Computer Graphics and Image Processing, 4, 248-270, 1975.

[DAV78] L.S. Davis and A. Rosenfeld, "Hierarchical Relaxation for Waveform Parsing," in Computer Vision Systems (A. Hanson and E. Riseman, Eds.), Academic Press, New York, 1978.

[EHR78] R.W. Ehrich and J.P. Foith, "Topology and Semantics of Intensity Arrays," in Computer Vision Systems (A. Hanson and E. Riseman, Eds.), Academic Press, New York, 1978.

[FIS78] M.A. Fischler, "On the Representation of Natural Scenes," in Computer Vision Systems (A. Hanson and E. Riseman, Eds.), Academic Press, New York, 1978.

[FEN75] H.Y. Feng and T. Pavlidis, "Decomposition of Polygons into Simpler Components: Feature Extraction for Syntactic Pattern Recognition," IEEE Trans. on Computers, C-24, 636-650, 1975.

[FRE76] E.C. Freuder, "Affinity: A Relative Approach to Region Finding," Computer Graphics and Image Processing, 5, 254-264, 1976.

[GOR74] W.J. Gordon and R.F. Riesenfeld, "B-Spline Curves and Surfaces," in Computer Aided Geometric Design (Barnhill and Riesenfeld, Eds.), Academic Press, New York, 1974.

[HAN74] A. Hanson and E. Riseman, "Preprocessing Cones: A Computational Structure for Scene Analysis," COINS Technical Report 74C-7, University of Massachusetts, September 1974.

[HAN75] A.R. Hanson, E.M. Riseman, and P. Nagin, "Region Growing in Textured Outdoor Scenes," Proc. of 3rd Milwaukee Symposium on Automated Computation and Control, 407-417, 1975.

[HAN76] A.R. Hanson and E.M. Riseman, "A Progress Report on VISIONS: Representation and Control in the Construction of Visual Models," COINS Technical Report 76-9, University of Massachusetts, 1976.

[HAN78] A.R. Hanson and E.M. Riseman, "VISIONS: A Computer System for Interpreting Scenes," in Computer Vision Systems (A. Hanson and E. Riseman, Eds.), Academic Press, New York, 1978.

[HAY74] K.C. Hayes, Jr., A.N. Shah, and A. Rosenfeld, "Texture Coarseness: Further Experiments," IEEE Trans. Systems, Man, and Cybernetics, 4, 467-472, 1974.

[HOR75] B.K.P. Horn, "Shape from Shading: A Method for Obtaining the Shape of a Smooth Opaque Object from One View," in The Psychology of Computer Vision (P. Winston, Ed.), McGraw-Hill, New York, 1975.

[KEL71] M.D. Kelly, "Edge Detection in Pictures by Computer Using Planning," Machine Intelligence, 6, 379-409, 1971.

[KEN76] J.R. Kender, "Saturation, Hue and Normalized Color: Calculation, Digitization Effects, and Use," Technical Report, Department of Computer Science, Carnegie-Mellon University, November, 1976.

[KIR71] R.A. Kirsch, "Computer Determination of the Constituent Structure of Biological Images," Computers and Biomedical Research, 4, 315-328, 1971.

[KLI76] A. Klinger and C.R. Dyer, "Experiments on Picture Processing Using Regular Decomposition," Computer Graphics and Image Processing, 5, 68-105, March 1976.

[KOH78] R. Kohler, COINS Technical Report, University of Massachusetts, in preparation.

[LEV78] M.D. Levine, "A Knowledge Based Computer Vision System," in Computer Vision Systems (A. Hanson and E. Riseman, Eds.), Academic Press, New York, 1978.

[MAR75] D. Marr, "Early Processing of Visual Information," AI Memo 340, Massachusetts Institute of Technology, 1975.

[NAG77] P.A. Nagin, A.R. Hanson and E.M. Riseman, "Region Extraction and Description Through Planning," COINS Technical Report 77-8, University of Massachusetts, May 1977.

[NAG78] P.A. Nagin, COINS Technical Report, University of Massachusetts, in preparation.

[OHL75] R. Ohlander, "Analysis of Natural Scenes," Ph.D. Thesis, Carnegie-Mellon University, April 1975.

[PAV75] T. Pavlidis, "Polygonal Approximations by Newton's Methods," Technical Report No. 194, Computer Science Laboratory, Princeton University, October 1975.

[PRA77] J.M. Prager, A.R. Hanson and E.M. Riseman, "Extracting and Labelling Boundary Segments in Natural Scenes," COINS Technical Report 77-7, University of Massachusetts, May 1977.

[PRA78] J.M. Prager, COINS Technical Report, University of Massachusetts, in preparation.

[PRI77] K. Price and R. Reddy, "Change Detection and Analysis in Multispectral Images," Proc. of 5th International Joint Conference on Artificial Intelligence, Cambridge, 619-625, 1977.

[RIS74] E.M. Riseman and A.R. Hanson, "The Design of a Semantically Directed Vision Processor," COINS Technical Report 74C-1, University of Massachusetts, January 1974.

[RIS77] E.M. Riseman and M.A. Arbib," Computational Techniques in the Visual Segmentation of Static Scenes," Computer Graphics and Image Processing, 6, 221-276, 1977.

[ROS71] A. Rosenfeld and M. Thurston, "Edge and Curve Detection for Visual Scene Analysis," IEEE Trans. Computers, 562-569, 1971.

[ROS76a] A. Rosenfeld and A.C. Kak, Digital Picture Processing, Academic Press, New York, 1976.

[ROS76b] A. Rosenfeld, R.A. Hummel and S.W. Zucker, "Scene Labelling by Relaxation Operations," IEEE Trans. Systems, Man, and Cybernetics, 6, 420-433, 1976.

[ROS77a] A. Rosenfeld, "Iterative Methods in Image Analysis," TR-517, Computer Science Center, University of Maryland, April 1977.

[ROS77b] A. Rosenfeld and L.S. Davis, "Iterative Histogram Modification," TR-519, Computer Science Center, University of Maryland, April 1977.

[SCH77] B.J. Schachter, L.S. Davis and A. Rosenfeld, "Some Experiments in Image Segmentation by Clustering of Local Feature Values," TR-510, Computer Science Center, University of Maryland, March 1977.

[SHI78] Y. Shirai, "Recognition of Real-World Objects Using Edge Cue," in Computer Vision Systems (A. Hanson and E. Riseman, Eds.), Academic Press, New York, 1978.

[SLO75] K.R. Sloan and R. Bajcsy, "A Computational Structure for Color Perception," Proc. of ACM75, Minneapolis, Minnesota, 1975.

[TAN75] S.L. Tanimoto and T. Pavlidis, "A Hierarchical Data Structure for Picture Processing," Computer Graphics and Image Processing, 2, 104-119, June 1975.

[TAN78] S.L. Tanimoto, "Regular Hierarchial Image and Processing Structures in Machine Vision," in Computer Vision Systems (A. Hanson and E. Riseman, Eds.), Academic Press, New York, 1978.

[TEN74] J.M. Tenenbaum, T.D. Garvey, S. Weyl, and H.D. Wolf, "An Interactive Facility for Scene Analysis Research," SRI Technical Note 87, Artificial Intelligence Center, Stanford Research Institute, 1974.

[TEN76] J.M. Tenenbaum and H.G. Barrow, "IGS: A Paradigm for Integrating Image Segmentation and Interpretation," Pattern Recognition and Artificial Intelligence (C.H. Chen, Ed.), Academic Press, 472-507, 1976.

[UHR72] L. Uhr, "Layered 'Recognition Cone' Networks That Preprocess, Classify, and Describe," IEEE Trans. Computers, 758-768, 1972.

[YAK73] Y. Yakimovsky and J.A. Feldman, "A Semantics-Based Decision Theory Region Analyzer," Proc. of 3rd International Joint Conference on Artificial Intelligence, Stanford University, 580-588, 1973.

[YAK76] Y. Yakimovsky, "Boundary and Object Detection in Real World Images," Journal of the ACM, 23, 599-618, 1976.

[YOR78a] B. York, "Shape Representation in the VISIONS System," COINS Technical Report, University of Massachusetts, in preparation.

[YOR78b] B. York, "Symbolic Classification of Primitive Two-Dimensional Shapes," COINS Technical Report, University of Massachusetts, in preparation.

[ZUC75] S. Zucker, A. Rosenfeld and L. Davis, "General Purpose Models: Expectations About the Unexpected," Proc. 4th International Joint Conference on Artificial Intelligence, 716-721, September 1975.

[ZUC76] S.W. Zucker, "Relaxation Labelling and the Reduction of Local Ambiguities," Proc. of 3rd International Joint Conference on Pattern Recognition, San Diego, 1976.

[ZUC77] S.W. Zucker, "Computing the Shape of Dot Clusters, I: Labelling Edge, Interior, and Noise Points," TR-543, Computer Science Center, University of Maryland, May 1977.