

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Technical Memorandum 33-709

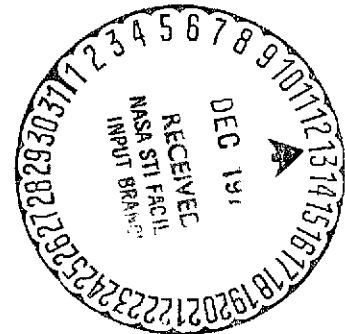
*Boundary and Object Detection in
Real World Images*

Yoram Yakimovsky

(NASA-CR-140828) BOUNDARY AND OBJECT
DETECTION IN REAL WORLD IMAGES (Jet
Propulsion Lab.) 30 p HC \$3.75 CSCL 20F

N75-12734

G3/74 03565
Unclas



JET PROPULSION LABORATORY
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA

November 15, 1974

Prepared Under Contract No. NAS 7-100
National Aeronautics and Space Administration

PREFACE

The work described in this report was performed by the Space Sciences Division of the Jet Propulsion Laboratory.

CONTENTS

I. Introduction	1
II. Definition of Terms	2
III. The Local Edge Detector	3
IV. Measuring Differences in Structure Between Two Neighborhoods . . .	3
V. Neighborhood Selection	10
VI. Locking on a Detected Edge	12
VII. Region Growing	13
VIII. Algorithm Description	14
IX. Simplification of the Result of Basic Region Growing	18
X. Growing Open Cracks into Closed Cracks	18
XI. Breaking a Region Into Two Around a Crack	19
XII. Merging Regions	20

FIGURES

1. Edge unit structure	23
2. Illustration of terms	23
3. Typical edges	24
4. Typical neighborhoods for edge detection	24
5. Extended neighborhoods set	25
6. An ideal edge value cross section	25
7. Region growing ambiguity example	26
8. Algorithm terms definition	26
9. The different region growing decisions	27
10. The three options to extend an open crack and the corresponding assumption on distributing	28

PRECEDING PAGE BLANK NOT FILMED

ABSTRACT

A solution to the problem of automatic location of objects in digital pictures by computer is presented. A self-scaling local edge detector which can be applied in parallel on a picture is described. Clustering algorithms and boundary following algorithms which are sequential in nature process the edge data to locate images of objects.

I. INTRODUCTION

A substantial amount of research was done in developing techniques for locating objects of interest automatically in digitized pictures. Drawing the boundaries around objects is essential for pattern recognition, tracing of objects in sequence of pictures for control systems, image enhancement, data reduction, and various other applications. References 1, 2, and 3 comprise a good survey of the research and application of image processing and picture analysis.

Most researchers of picture analysis assumed that (1) the image of an object is more or less uniform or smooth in its local properties (that is, illumination, color, and local texture are smoothly changing inside the image of an object) and (2) there is detectable discontinuity in local properties between images of two different objects. We will adopt these two assumptions in this paper and assume no textural image (see Ref. 4 for an example of texture image analysis which does not make these assumptions).

The work on automatic location of objects in digitized images was split into two branches: (1) the edge detection and edge following vs (2) the region growing. The edge detection meant applying in different points over the picture local independent operators to detect edges and then using algorithms to trace the boundaries by following the local edge detected. A recent survey of literature in this area is given in Ref. 5. The region growing approach was to use various clustering algorithms to grow regions of almost uniform local properties in the image. (See Refs. 6-9 for typical applications.) More detailed references will be given later.

In this paper the two approaches are combined to complement each other. The end result is a more powerful mechanism to do the job of picture segmentation. We developed a new edge detector and combined it with new region growing techniques to locate objects and thereby resolved the confusion that has resulted for regular edge following when more than one isolated object on a uniform background is in the scene (see Ref. 10).

The contributions of this report are the following:

- (1) A new and "optimal" (given certain assumption) edge detector is presented.

- (2) A simple one-pass algorithm to do region growing is presented which utilizes the edge detector output.
- (3) The application of path generator algorithms and "shortest path algorithms" to do the boundary following so as to close open edge lines into boundaries around regions is discussed.
- (4) Special-purpose region growing intended to close open edges (cracks) is described.
- (5) A special clustering algorithm which simplifies the region structure resulting from application of (1) through (5) is presented.

II. DEFINITION OF TERMS

The input is expected to be in matrix form $\vec{V}(i,j)$ $i = 1, \dots, N$ $j = 1, \dots, M$, where \vec{V} is a vector in R^n , n is a function of the sensory system, usually 1 (gray level picture), or 3 (color or x, y, z coordinates of surface in the scanning direction), or 6 (color and 3-D information). An edge unit separates two adjacent matrix points; that is, an edge unit is between (i,j) and $(i+1, j)$ or between (i,j) and $(i, j+1)$ for some i,j , (see Fig. 1).

An edge unit is usually adjacent on both ends to other edge units. There are 64 combinations of edge units continuing an edge unit since each of the edge units $e_1, e_2, e_3, e_1^1, e_2^1, e_3^1$ in Fig. 1 may exist or not.

Two points on the grid (I, J) and (K, L) are said to be in the same region if there is a path sequence $(i_1, j_1), \dots, (i_n, j_n)$ such that $i_1 = I, j_1 = J, i_n = K$ and $j_n = L$, where (i_m, j_m) is adjacent to (i_{m+1}, j_{m+1}) for $m = 1, \dots, n-1$ and there is no edge unit between the two. A region will be a maximum set of points satisfying that property.

An edge-line (or an edge) between region R_1 and region R_2 is the maximal sequence of adjacent edge units such that each edge unit in the sequence is between two matrix points, one belonging to R_1 and the other to R_2 . It is possible that an edge line is inside a region ($R_1 = R_2$).

An edge line which is between two different regions is called a boundary. An edge line which is inside a region is called a crack. An open crack is a crack in which at least one end terminates without connecting to any edge line. A closed crack is one which terminates at both ends on another edge

line. For instance, cracks will appear when an object is smoothly disappearing into the background on one side and has detectable discontinuity on the other side (Fig. 2).

Using the above definitions, this report presents an edge detector which detects edge units in parallel locally on the whole image. Then a region grower which results in the grouping of matrix points into regions and edge units into boundaries and cracks is presented. A local region grower which tries to break a region with a crack in it into two regions for which the crack is part of the common boundary is then presented. Alternatively, an open-crack-extending algorithm is suggested to connect the open edge unit of the crack to another edge line.

III. THE LOCAL EDGE DETECTOR

The edge operator is a detector of local discontinuity in an image. When applied between two adjacent points such as (i,j) and $(i+1,j)$, it should return a value which will measure the confidence that there is an edge between (i,j) and $(i+1,j)$. Since we work with noisy input to achieve reliability, the operator must look at two 2-dimensional (2-D) neighborhoods N_1 and N_2 to obtain a reliable value. Neighborhood N_1 will include (i,j) and a few adjacent points; N_2 includes $(i+1,j)$ and a few adjacent points; and $N_1 \cap N_2 = \emptyset$. As a result the value returns will measure the confidence that the neighborhoods belong to images of different objects.

Edge detection is actually composed of three components: (1) choosing the proper neighborhoods, (2) the measurements of differences between image structures in the two neighborhoods, and (3) locking on the exact position of the edge. Discussion of each of these steps follows.

IV. MEASURING DIFFERENCES IN STRUCTURE BETWEEN TWO NEIGHBORHOODS

Any techniques which measure structural differences must make some assumption (explicitly or implicitly) on the structure of an edge vs the area inside the image of a region. Binford and Hershkowitz (referred to in Ref. 5) suggested three possible ideal edges defined by the reading profile on a normal-to-the-edge line (Fig. 3).

All of these idealized edges are in reality washed with white noise on both sides, where the noise is the result of both hardware noise and surface irregularities. Basically, the decision needed to be made is between two hypotheses:

H_0 : The readings in N_1 and N_2 are taken from the same object.

H_1 : The readings in N_1 are taken from one object and in N_2 from another object.

Neighborhoods N_1 and N_2 are the neighborhoods mentioned in the previous section, and the decision as to how to choose them will be described in the next section.

An optimal (best for its size) decision between H_0 and H_1 will utilize the maximum likelihood ratio as follows: Let P_0 be the maximum likelihood estimate of the structure (reading in N_1 and N_2), given that H_0 is true, and let P_1 be the maximum likelihood estimate of the structure, assuming H_1 is true. Then,

Choose H_1 when $\frac{P_1}{P_0} > K$

Choose H_0 when $\frac{P_1}{P_0} < K$

With probability γ , choose H_0 and with probability $1 - \gamma$, choose

$$H_1 \text{ when } \frac{P_1}{P_0} = K \quad 0 \leq \gamma \leq 1$$

This decision will be optimal for its size (see the Neyman-Pearson lemma in Ref. 11, page 201); hence, if the structure assumptions are valid, we have an ideal edge detection, given only readings in N_1 and N_2 . (We will deal with gaussian probabilities; hence we will ignore $P_0/P_1 = K$.) The conclusion is that P_1/P_0 is the best measure of the edge strength. Following are two examples of applying these principles to edges of types (a) and (b) in Fig. 3.

EXAMPLE 1

Assume that the edges and surfaces will be of type A, with added white noise which is object-dependent. Then H_0 and H_1 will become

H_0 : The readings in both N_1 and N_2 are taken from the same normal distribution $N(\mu_0, \sigma_0)$ with unknown μ_0, σ_0 .

H_1 : The readings on N_1 are taken from normal distribution $N(\mu_1, \sigma_1)$; and the readings on N_2 are taken from normal distribution $N(\mu_2, \sigma_2)$; (μ_1, σ_1) need not be equal to (μ_2, σ_2) .

To apply the maximum likelihood ratio principle we need to find a maximum likelihood estimate for (μ_0, σ_0) , (μ_1, σ_1) and (μ_2, σ_2) . Given (x_1, \dots, x_n) readings taken from a normal distribution with unknown (μ, σ) , the maximum likelihood estimates are

$$\mu = \frac{\sum_{i=1}^n x_i}{n}$$

$$\sigma^2 = \frac{\sum_{i=1}^n (x_i - \mu)^2}{n}$$

$$P_{\max} = P_{(\mu, \sigma)}(x_1, \dots, x_n)$$

$$= \frac{1}{(\sqrt{2\pi \cdot \sigma})^n} \cdot e^{-\frac{1}{2\sigma^2} \cdot \sum_{i=1}^n (x_i - \mu)^2}$$

$$= \frac{1}{(\sqrt{2\pi} \cdot \sigma)^n} \cdot e^{-\frac{n\sigma^2}{2\sigma^2}}$$

$$= \left(\frac{1}{\sqrt{2\pi}} \cdot e^{-\frac{n}{2}} \right) \cdot \frac{1}{\sigma^n}$$

Hence, if the readings on N_1 are (x_1, \dots, x_m) and on N_2 (y_1, \dots, y_n) , then, on N_1 ,

$$\mu_1 = \frac{\sum_{i=1}^m x_i}{m}$$

$$\sigma_1^2 = \frac{\sum_{i=1}^m (x_i - \mu_1)^2}{m}$$

$$P_1 = \left(\frac{1}{(2\pi)^{\frac{m}{2}}} \cdot e^{-\frac{m}{2}} \right) \cdot \frac{1}{\sigma_1^m}$$

On N_2 ,

$$\mu_2 = \frac{\sum_{i=1}^n y_i}{n}$$

$$\sigma_2^2 = \frac{\sum_{i=1}^n (y_i - \mu_2)^2}{n}$$

$$P_2 = \left(\frac{1}{(2\pi)^{n/2}} \cdot e^{-\frac{n}{2}} \right) \cdot \frac{1}{\sigma_2^n}$$

and on N_1 combined with N_2 ,

$$\mu_0 = \frac{m\mu_1 + n\mu_2}{m+n}$$

$$\sigma_0^2 = \frac{m\sigma_1^2 + n\sigma_2^2 + m(\mu_0 - \mu_1)^2 + n(\mu_0 - \mu_2)^2}{m+n}$$

$$P_0 = \frac{1}{(2\pi)^{\frac{m+n}{2}}} \cdot e^{-\frac{m+n}{2}} \cdot \frac{1}{\sigma_0^{m+n}}$$

$$\frac{P_1^2 \cdot P_2^2}{P_0^2} = \frac{\left(\sigma_0^2\right)^{m+n}}{\left(\sigma_1^2\right)^m \cdot \left(\sigma_2^2\right)^n}$$

(It is convenient to work with σ^2 since it saves computation of square roots.)

An "almost always" good threshold in this example was $K^2 = 25$. That is, if $P_1^2/P_0^2 \geq 25$, decide that there is an edge; otherwise there is no edge. Note that the threshold is self-scaling. In noisy or highly textured areas it will in effect require a higher step for an edge, and in smooth areas it will require a lower step. In practice, we also always forced our σ^2 to be greater than 0.25 since all our readings were digitized, which meant a ± 0.5 random error in the readings.

At this point it may be worthwhile to compare our approach with that of Ref. 12. Both try to use a maximum likelihood ratio to compute scores for an edge. But while we have a simple model and a practical way of computing the confidence, Ref. 12 assumes a priori deterministic classification of all possible idealized noise-free structures to edges and no edges. Then, for a given reading structure, the noise assumption is used to compute the probability of all

idealized structures that could have caused the readings. These probabilities are used to decide whether the readings represent on edge or not.

EXAMPLE 2

Here we assume that each matrix point $\vec{V}(i,j)$ is a 3-dimensional vector (x,y,z) . Actually the raw readings are just distance $R(i,j)$, but to avoid a strong dependency on the sensory position, (i,j,R) are used to compute (x,y,z) . This is the form of input read from a device which measures distances to surfaces (such as radar or devices which measure the time of flight of laser beams to an object). The i,j corresponds to vertical and horizontal steps in the scanning angle. The two adjacent neighborhoods on the matrix N_1 and N_2 have readings $(x_1, y_1, z_1), \dots, (x_n, y_n, z_n)$ in N_1 and $(x_1^1, y_1^1, z_1^1), \dots, (x_m^1, y_m^1, z_m^1)$ in N_2 . We assume that objects are almost planar locally with added white noise. That is, if we read $(x_1, y_1, z_1), \dots, (x_n, y_n, z_n)$ in a small neighborhood on an object we have a, b, c, d, σ such that

$$a^2 + b^2 + c^2 = 1$$

and

$$ax_i + by_i + cz_i + d + N(0, \sigma) = 0 \quad i = 1, \dots, N$$

With this assumption the edge detection decision will be a choice between H_0 and H_1 .

H_0 : The readings in the two neighborhoods are taken from the same plane. That is the readings on both N_1 and N_2 satisfy for some $(a_0, b_0, c_0, d_0, \sigma_0)$

$$a_0x + b_0y + c_0z + d_0 + N(0, \sigma_0) = 0$$

where

$$a_0^2 + b_0^2 + c_0^2 = 1$$

for all (x,y,z) readings in N_1 and N_2 .

H_1 : There are two not necessarily equal planar fits for the readings on N_1 and on N_2 . That is, there are $(a_1, b_1, c_1, d_1, \sigma_1)$ for N_1

and $(a_2, b_2, c_2, d_2, \sigma_2)$ for N_2 such that

$$a_1^2 + b_1^2 + c_1^2 = 1$$

$$a_2^2 + b_2^2 + c_2^2 = 1$$

$$i = 1, \dots, n \quad a_1 x_i + b_1 y_i + c_1 z_i + d_1 + N(0, \sigma_1) = 0$$

$$i = 1, \dots, m \quad a_2 x_i + b_2 y_i + c_2 z_i + d_2 + N(0, \sigma_2) = 0$$

To apply the Neyman-Pearson principle for this case we want to find maximum likelihood estimates. Maximum likelihood estimates a_1, b_1, c_1, d_1 will be

$$V_1 = \sum_{i=1}^n (a_1 x_i + b_1 y_i + c_1 z_i + d_1)^2 = \min_{\substack{a, b, c, d \\ a^2 + b^2 + c^2 = 1}} \sum_{i=1}^n (a x_i + b y_i + c z_i + d)^2$$

and

$$\sigma_1^2 = V_1$$

Solving for the optimal (a_1, b_1, c_1, d_1) is a relatively straightforward process. Once they are found, the maximum likelihood estimate for N_1 will be

$$P_1 = \frac{1}{\sqrt{2\pi} \cdot \sigma_1^n} \cdot e^{-\frac{n}{2}}$$

Hence, we have the expression which tests for an edge. It is of the following form: If

$$\frac{V_0^{m+n}}{V_1^n \cdot V_2^m} \geq K^2$$

decide for H_1 , otherwise H_0 .

Note that (x, y, z) may be replaced by (i, j, g) in regular black and white pictures, in which case we will have a regular picture edge operation which will be able to handle edges of type B in Fig. 3.

V. NEIGHBORHOOD SELECTION

In the previous discussion on decision criteria we deliberately left out the question of how to choose the test neighborhoods. This is another variant of the properties that we want the edges to have. The edge value for a vertical edge between two horizontally adjacent points is taken to be the strongest case for an edge computed on the four pairs of neighborhoods (a) through (d) in Fig. 4. Taking the maximum of the maximum likelihood ratio estimate for an edge among the four values computed for the four neighborhoods is similar to the approach advocated in Ref. 13.

A completely symmetric configuration is used to measure the confidence value of a horizontal edge unit between two vertically adjacent points. The choice of neighbors is of an experimental nature, and it worked for our problems. Other problem-dependent neighborhood choices are possible, and they will work for the specific edge structure in mind (see examples in Fig. 5). In choosing the size of a neighborhood, a reasonable balance between noise and size of object should be achieved. The bigger the neighborhoods the less sensitive to noise the decision will be, but the small objects may be lost.

At this point it is worthwhile referring to the edge detector developed by Hueckel (Ref. 14). He found an elegant technique to compute parameters for step function:

$$\text{STEP}_{a,b,c,d,e}^{(i,j)} = \begin{cases} d & ai + bj \geq c \\ e & ai + bj < c \end{cases}$$

For a disk

$$(i,j) \in \text{DISK}(i_0, j_0, \gamma) \triangleq \left\{ (i,j) \mid (i - i_0)^2 + (j - j_0)^2 \leq \gamma \right\}$$

which minimizes for a given signal $g(i,j)$ in the disk the quantity

$$\sum_{(i,j) \in \text{DISK}} \left(g(i,j) - \text{STEP}_{a,b,c,d,e}^{(i,j)} \right)^2$$

over all possible step functions. He took the parameters of a,b,c,d,e to be the parameters of the "best possible" edge passing through the disk. This measure of edge quality is clearly different from ours. Since our measure of edge strength is more complicated, it is unlikely that an elegant and simple way of finding optimal edge through a disk using our measure of edge strength is achievable. However, given a suggested edge structure, our approach can be used immediately to provide a model-driven confidence evaluation in the existence of the suggested edge. For the suggested (a,b,c,d,e) edge parameter let

$$N_2 = \sum_{\substack{ai + bj \geq c \\ (i,j) \in \text{DISK}}} 1$$

$$\mu_2 = d$$

$$\sigma_2^2 = \frac{\sum_{\substack{ai + bj \geq c \\ (i,j) \in \text{DISK}}} (g(i,j) - d)^2}{N_2}$$

$$N_1 = \sum_{\substack{ai + bj < c \\ (i,j) \in \text{DISK}}} 1$$

$$\mu_1 = e$$

$$\sigma_1^2 = \frac{\sum_{\substack{ai + bj < c \\ (i,j) \in \text{DISK}}} (g(i,j) - e)^2}{N_1}$$

$$N_0 = N_1 + N_2$$

$$\mu_0 = (N_2 \cdot \mu_2 + N_1 \cdot \mu_1) / (N_1 + N_2)$$

$$\sigma_0^2 = \frac{\sum_{(i,j) \in \text{DISK}} (g(i,j) - \mu_0)^2}{N_0}$$

Then,

$$\text{STRENGTH} = \frac{\binom{\sigma_0^2}{N_0}}{\binom{\sigma_1^2}{N_1} \binom{\sigma_2^2}{N_2}}$$

VI. LOCKING ON A DETECTED EDGE

Computing the edge value is not usually sufficient to decide where to put the edges. The values that are computed usually look like the ones in Fig. 6.

One way of forcing the edge to be well defined is to constrain it to be a local maximum in addition to having a confidence value higher than a certain threshold. This is, of course, extremely important for locking on the center of the edge. Usually there is still some local ambiguity on the location of the edge, and for many practical reasons it is better to treat the area around an edge as ambiguous. The source of problems here is that, because of computing time constraints, it was impossible to find a global optimum for edge lines using all available data, and it was necessary to use only local information for evaluating edge units in this level. In our system, the decision as to where exactly to put the edge was left for the region grower (see below). To demonstrate the possible 2-D ambiguity, see Fig. 7.

The search for a maximum may be used for special-purpose edge detection. For instance, if we look only for one dark stripe crossing a white background, forcing the edge to be the absolute maximum or minimum on a horizontal line in the image (keeping track of signs of change) will supply the appropriate pair of edges.

VII. REGION GROWING

The output of an application of an edge detector results in two new matrices in addition to the matrix $\vec{V}(i,j)$ of raw data. The first is $EV(i,j)$, which is the measure of the confidence that there is an edge unit between (i,j) and $(i,j+1)$; the second is $EH(i,j)$, which measures the confidence that there is an edge unit between (i,j) and $(i+1,j)$. $EV(i,j)$ and $EH(i,j)$ may include extra bits as determined by the direction of the change on that suggested edge unit.

This output as it stands is not sufficient for application of pattern recognition and various picture quantitative analysis tasks, since outlines of objects are needed in order to recognize features. Hence a region grower which will outline objects is needed. A straightforward approach is to classify as edge units all edge units where $EV(i,j)$ or $EH(i,j)$ are of confidence value greater than some threshold T and which are local maxima. A point (i_0,j_0) is a local maximum if $EV(i_0,j_0) \geq EV(i_0,j_0+1)$ and $EV(i_0,j_0) \geq EV(i_0,j_0-1)$ or $EH(i_0,j_0) \geq EH(i_0+1,j_0)$ and $EH(i_0,j_0) \geq EH(i_0-1,j_0)$. Unfortunately, this straightforward approach fails. Indeed, it creates an excellent display of boundaries for the viewer, but as a result of image irregularities too many cracks are generated and too many skinny regions appear on boundaries of objects. Because of that, a sequence of algorithms is called to utilize more global structure to find exact positions of boundary lines and eliminate most cracks and regions which are too small to be of interest.

We start by describing a one-pass algorithm which parses the edge data into data structures of regions, boundaries, closed cracks, and open cracks and creates, as byproducts, two arrays, $FH(i,j)$ and $FV(i,j)$, where $FH(i,j)$ means that the program puts an edge unit between $(i-1,j)$ and (i,j) , and $FV(i,j)$ means an edge unit between (i,j) and $(i,j-1)$.

To ease the description of the decision mechanism for putting edges, we need to define a few new terms. Let $T > 0$ be the edge confidence threshold; then,

- (1) 'd' is the distance between two adjacent grid points. It will be
- $$d((i,j), (i-1,j)) \triangleq d((i-1,j), (i,j)) \triangleq \begin{cases} \text{if } EH(i,j) \leq T \\ \text{then } 0, \text{ else } EH(i,j) \end{cases}$$
- $$d((i,j), (i,j-1)) \triangleq d((i,j-1), (i,j)) \triangleq \begin{cases} \text{if } EV(i,j) \leq T \end{cases}$$

then 0, else EV(i,j)

- (2) Reg(i,j) will be the region to which the point (i,j) belong.
(Reg(i,j) is not defined to all points until the program is finished.)

$$(3) \quad \text{Val}(i,j) = \text{Min} \{ \underbrace{d((i,j), (k,l))}_{\substack{|i-k| + |j-l| = 1 \\ \text{No edge unit between} \\ (i,j) \text{ and } (k,l)}} \}$$

This value will be $+\infty$ if (i,j) is the only point in its region.

$$(4) \quad \text{Val}(\underbrace{\text{Reg}_1}_{\substack{(i,j) \\ \text{Reg}(i,j) = \text{Reg}_1}}) = \text{Min}(\text{Val}(i,j))$$

- (5) A point P will be the minimum point for its region if
 $\text{Val}(P) = \text{Val}(\text{Reg}(P))$

The algorithm is designed so that at each state there is always a non-decreasing path from each minimum of any region to any other point in the region and the path enters that point from its minimum direction.

That is, if P and Q are two points such that

$\text{Reg}(P) = \text{Reg}(Q)$ and $\text{Val}(P) = \text{Val}(\text{Reg}(P))$, then
there is a path (x_1, x_2, \dots, x_n) such that

- (a) $x_1 = P, \quad x_n = Q$
- (b) $\text{Reg}(x_i) = \text{Reg}(P), \quad i = 1, \dots, n$
- (c) x_i adjacent to $x_{i+1}, \quad d(x_{i+2}, x_{i+1}) \geq d(x_{i+1}, x_i)$
- (d) $d(x_n, x_{n-1}) = \text{Val}(x_n)$

We say if such a path exists that Q is reachable from P.

That is, two points are in the same region if you can get from one to the other in a path which does not cross a ridge of edge values.

VIII. ALGORITHM DESCRIPTION

The program scans the image from left to right, line by line. That is, the scanning is such that when point (i,j) is processed, the program already worked on all points (i_1, j_1) such that $(j_1 < j)$ or $(j = j_1 \text{ and } i_1 < i)$.

Assume the program is processing point (i,j) .

Let D_1 be a Boolean variable set to true if this program is not going to put an edge unit between (i,j) and $(i,j-1)$, and false otherwise, and let D_2 be a Boolean variable set to true if the program is not going to put an edge unit between (i,j) and $(i-1,j)$, and false otherwise. Let $R_1 = \text{Reg}(i,j-1)$ and $R_2 = \text{Reg}(i-1,j)$ (see Fig. 8).

The decision on the values of D_1 and D_2 is described by the following ALGOL-like program:

Begin

Boolean Good-Down₁, Bad-Down₁, Up₁, Good-Down₂, Bad-Down₂, Up₂;

Good-Down₁ \leftarrow if $d((i,j), (i,j-1)) \leq \text{Val}(i,j-1) \wedge \text{Val}(i,j-1) \leq \text{Val}(R_1)$
then TRUE
else FALSE;

Comment: Good-Down₁ is true if point (i,j) is going to become a new minimum for R_1 (the region above). And it is adjacent to an old minimum; hence, any point of R_1 reachable from the old adjacent minimum will be reachable from the new;

Bad-Down₁ \leftarrow if $d((i,j), (i,j-1)) < \text{Val}(i,j-1) \wedge (\text{Val}(i,j-1) > \text{Val}(R_1))$
then TRUE
else FALSE;

Comment: This variable is true if (i,j) is not reachable from all minima of R_1 going through $(i,j-1)$;

Up₁ \leftarrow if $d((i,j), (i,j-1)) \geq \text{Val}(i,j-1)$
then TRUE
else FALSE;

Comment: This variable is true if point (i,j) is reachable from any minima of R_1 by continuing the path that leads from that minimum to $(i,j-1)$;

Good-Down₂ \leftarrow if $d((i,j), (i-1,j)) \leq \text{Val}(i-1,j) \wedge (\text{Val}(i-1,j) \leq \text{Val}(R_2))$
then TRUE
else FALSE;

Comment: This variable is true if point (i,j) is going to be a new minimum for R_2 (the region minimum, to the side) and is adjacent to an old

minimum of R_2 ; hence any point reachable from the adjacent old minimum will be reachable from (i,j) ;

```
Bad-Down2 ← if d((i,j), (i-1,j)) < Val(i-1,j) ∧ Val(i-1,j) > Val( $R_2$ )
then TRUE
else FALSE;
```

Comment: This variable is true if (i,j) is not reachable from all minima of R_2 through $(i-1,j)$;

```
Up2 ← if d(i,j), (i-1,j) ≥ Val(i-1,j)
then TRUE
else FALSE;
```

Comment: This variable is true if point (i,j) is reachable from any minima of R_2 by continuing the path that leads from that minimum to $(i-1,j)$;

```
If Good-Down1 ∧ Good-Down2 then D1 ← D2 ← true
else
```

```
If Good-Down1 ∧ Bad-Down2 then begin D1 ← true; D2 ← false; end
else if Good-Down1 ∧ Up2 then begin
```

```
    if d((i,j), (i,j-1)) ≥ d(i,j), (i-1,j))
    then D1 ← D2 ← true
    else begin D1 ← true;          D2 ← false; end;
           end
```

```
else if Bad-Down1 then begin if Good-Down2 V Up2 then begin
```

```
    D1 ← false
    D2 ← true
    end
    else begin
    D1 ← false
    D2 ← false; end
    end
```

```
else if Up1 ∧ Good-Down2 then begin
```

```
    if d((i,j), (i-1,j)) ≥ d((i,j), (i,j-1))
    then D1 ← D2 ← true
    else begin D2 ← true;          D1 ← false; end
           end
```

```
else if Up1 ∧ Up2 then, if  $R_1 = R_2$  then D1 ← D2 ← true else
    if d((i,j), (i-1,j)) ≥ d((i,j), (i,j-1))
```

```

        then begin  $D_1 \leftarrow \text{true};$      $D_2 \leftarrow \text{false};$  end
        else begin  $D_1 \leftarrow \text{false};$    $D_2 \leftarrow \text{true};$  end
    end
    Comment: Only one of  $D_1$  and  $D_2$  can be true; otherwise we cannot
    guarantee entrance through minimum value from all minima of both  $R_1$ 
    and  $R_2$ ;
else if  $Up_1 \wedge \text{Bad-Down}_2$  then begin  $D_1 \leftarrow \text{true};$   $D_2 \leftarrow \text{false};$  end
     $Val(i, j) \leftarrow \infty;$ 
    if  $D_1$  then begin
         $Val(i, j-1) \leftarrow \text{Min}(d((i, j), (i, j-1)), Val(i, j-1));$ 
         $Val(i, j) \leftarrow d((i, j), (i, j-1));$ 
         $Val(R_1) \leftarrow \text{Min}(Val(R_1), Val(i, j));$ 
    end;
    if  $D_2$  then begin
         $Val(i-1, j) \leftarrow \text{Min}(d((i, j), (i-1, j)), Val(i-1, j));$ 
         $Val(i, j) \leftarrow \text{Min}(Val(i, j), d((i, j), (i-1, j)));$ 
         $Val(R_2) \leftarrow \text{Min}(Val(R_2), Val(i, j));$ 
    end;
    If not  $(D_1 \vee D_2)$  then  $Val(Reg(i, j)) \leftarrow \infty;$ 

```

The e_1 and e_2 (see Fig. 8) may exist or not, and as a result there are four starting conditions. The program may put D_1 , D_2 , D_1 and D_2 or none of them, and hence, there are 16 cases in a point. (See Fig. 9 for a brief description of the different cases.)

Merging of two regions may always result in transformation into a crack of a previously common boundary of the two regions. In general, each operation of the region grower is fairly elaborate: more than meets the eye. The data structure used is not described in this paper, but it is essentially the same data structure described in Ref. 9, with slight modification to include edge line representation through chain encoding.

This one-pass algorithm is local and requires relatively small core resident data. However, it does not create maximal regions with respect to our criteria of path connectivity and reachability. The reason is the possible asymmetry of the distance function. On the other hand, it is relatively simple and fast when other algorithms are considered. The maximality problem may

be easily corrected if backup is allowed. Note also that, in fact, the threshold T plays a very small role in defining the output of the algorithm.

IX. SIMPLIFICATION OF THE RESULT OF BASIC REGION GROWING

There are two straightforward options for simplifying the output of the one-pass region grower: (1) take all regions that are too small to be interesting and melt them into their closest neighbor (the distance between two regions will be defined later in the paper); (2) take all short cracks which are weak (strength of the edge line will be defined later) and delete them. Of course, the threshold below which a crack is weak and a region is small is a function of how much we want to elaborate the task of the image analysis and is defined heuristically. In fact, in the current implementation all cracks are deleted since the edge operator was sensitive enough for our purposes.

X. GROWING OPEN CRACKS INTO CLOSED CRACKS

One possible way of closing open cracks is to grow them in length from their open end until the extended edge line meets already existing edge lines and closes. On the open end in each step there are three choices as to where to extend the edge line: go straight ahead, turn left, or turn right. The decision as to which direction to take will be to minimize the cost of closing the open crack, where the cost is defined heuristically. One possible choice is as follows:

Given the original crack, define two distributions which will describe the properties on either side of the crack, P_{D_1} and P_{D_2} . The cost for adding an edge unit will be the maximum likelihood ratio between the two assumptions.

H_0 : the two sides of the edge unit belong to the same side of the crack (the best choice between D_1 on both sides of the extension and D_2 on both sides).

H_1 : there is a different distribution on either side chosen according to geometrical constraint (Fig. 10).

$$\text{Cost} = P_{H_0} / P_{H_1}$$

Note that since the cost function is additive it can be used in conjunction with the shortest path algorithm (Ref. 15, Ch. 3) to find the nearest (least expensive) path to a closing edge unit. Reference 16 describes another heuristic of line expanding which may be applicable to our case.

XI. BREAKING A REGION INTO TWO AROUND A CRACK

An alternative approach to breaking a region into two regions so as to make the crack into a part of a boundary is to use special-purpose region growing. Assume that there is a crack in a regular gray level picture (that $\vec{V}(i,j) \in R_1$), with readings with mean μ_1 and variance σ_1^2 on a small neighborhood on one side of the crack and mean μ_2 and variance σ_2^2 on the other side. Assume that the crack is inside region R; then we can break the points in R into two classes, C_1 and C_2 :

$$C_1 \triangleq \left\{ (i,j) \mid \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{1}{2}\left(\frac{V(i,j) - \mu_1}{\sigma_1}\right)^2} \leq \frac{1}{\sqrt{2\pi}\sigma_2} \cdot e^{-\frac{1}{2}\left(\frac{V(i,j) - \mu_2}{\sigma_2}\right)^2} \right\}$$

$$C_2 \triangleq R - C_1$$

Then in some sense we would expect C_1 to be on the first side of the crack and C_2 on the second side of the crack. Unfortunately, it may turn out that C_1 or C_2 are not path-wise connected. As a result, one of the connected components which border on the crack should be picked out. A more heuristic approach is to grow a region around each of the two sides of the crack, and to stop when a new point has a neighborhood which is more likely to belong to the other side. Then take the smaller of the two regions resulting and make it C_1 ; then C_2 will be

$$C_2 = R - C_1$$

This algorithm can be used also to allow flexible human interaction in analyzing the scene.

XII. MERGING REGIONS

The basic region grower utilized local detection procedures. Better decisions are achievable (at least theoretically) by using more global information. The problem is how to allow this additional information and still keep the program lean and fast. Research in that area was reported (Ref. 16). Basically, our approach is to be oversensitive on the local pass and as a result to over-segment the picture. But then we take the data output (which is reduced data) and simplify it. We take pairs of regions with common boundaries and merge them into one. In order to do that reliably, a confidence value which measures the confidence that the pair of regions are different is computed, and iteratively we pick the pair of regions with the lowest confidence of being different in the current structure, merge them, and update the structure. The confidence is computed as the product of two components: (1) edge line strength (on the common boundary of the two regions) and (2) the difference of the properties inside the region. Both of these values are computed on the basis of assumptions similar to those used in the edge confidence evaluation. For instance, if we assume gray level readings, at each point (i,j) the value is a positive integer value. Along the edge line take two small neighborhoods on the two sides (like a 4-point-wide stripe on each side of the edge) and assume that the readings in one neighborhood are

$$(x_i)_{i=1}^n$$

and

$$(x_i')_{i=1}^m$$

in the other. Then the edge strength will be the ratio between the maximum likelihood estimate that $(x_i)_{i=1}^n$ $(x_i')_{i=1}^m$ are from two different normal distributions to the maximum likelihood estimate that $(x_i)_{i=1}^n$ $(x_i')_{i=1}^m$ are taken from the same normal distribution. The computation technique for the values is the same as that used in the edge evaluation model 1. This value gives the boundary

strength evaluation; a similar value is computed on the basis of distribution in each region, which gives the difference in properties of the two regions.

REFERENCES

1. Rosenfeld, A. , "Picture Processing by Computer," Computing Survey, Vol. 1, pp. 147-176, September 1969.
2. Rosenfeld, A. , Progress in Picture Processing: 1969-1971, C. S. TR-176, University of Maryland, January 1972.
3. Rosenfeld, A. , Picture Processing: 1972, C. S. TR-217, University of Maryland, January 1973.
4. Bajcy, R. , Computer Identification of Textured Scene, AIM-180, STAN-CS-72-321, Stanford University, 1972.
5. Davis, L. , A Survey of Edge Detection Techniques, C. S. TR-273, University of Maryland, November 1973.
6. Brice, C. , and Fennema, C. , "Scene Analysis Using Regions," Journal of Artificial Intelligence, Vol. 1, pp. 205-226, 1970.
7. Barrow, H. , and Popplestone, R. , "Relational Description in Picture Processing," Machine Intelligence, Vol. 6, pp. 377-396.
8. Harlow, C. A. , and Eisenbeis, S. A. , "The Analysis of Radiographic Images," IEEE Transactions on Computers, Vol. L-22, pp. 678-689, 1973.
9. Yakimovsky, Y. , Scene Analysis Using a Semantic Base for Region Growing, STAN-CS-73-380, Stanford University, June 1973.
10. Pingle, K. , and Tenenbaum, J. , "An Accommodating Edge Follower," Proceedings of the International Joint Conference on Artificial Intelligence, September 1971.
11. Ferguson, T. S. , Mathematical Statistics, a Decision Analysis Approach, Academic Press, 1967.
12. Griffith, A. , "Mathematical Models for Automatic Line Detection," Journal of the Association of Computing Machinery, Vol. 20, pp. 62-80, 1973.
13. Carton, E. J. , et al. , Some Basic Edge Detection Techniques, TR-277, University of Maryland Computer Science Center, December 1973.
14. Hueckel, M. H. , "A Local Visual Operator Which Recognizes Edges and Lines," Journal of the Association of Computing Machinery, Vol. 20, pp. 634-647, 1973.
15. Nilsson, N. , Problem Solving Methods in Artificial Intelligence, McGraw-Hill, 1971.
16. Montanari, U. , "On Optimal Detection of Curves in Noisy Pictures," Communications of the Association of Computing Machinery, Vol. 14, pp. 335-345, 1971.

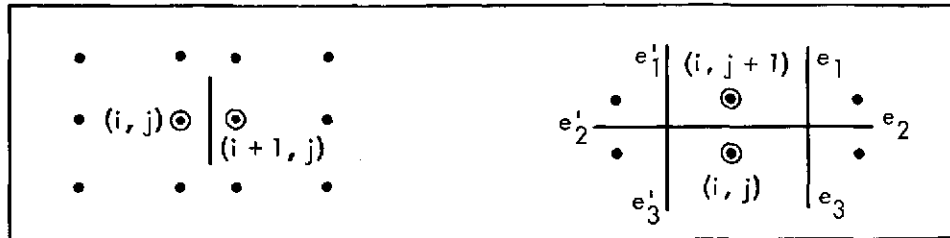


Fig. 1. Edge unit structure

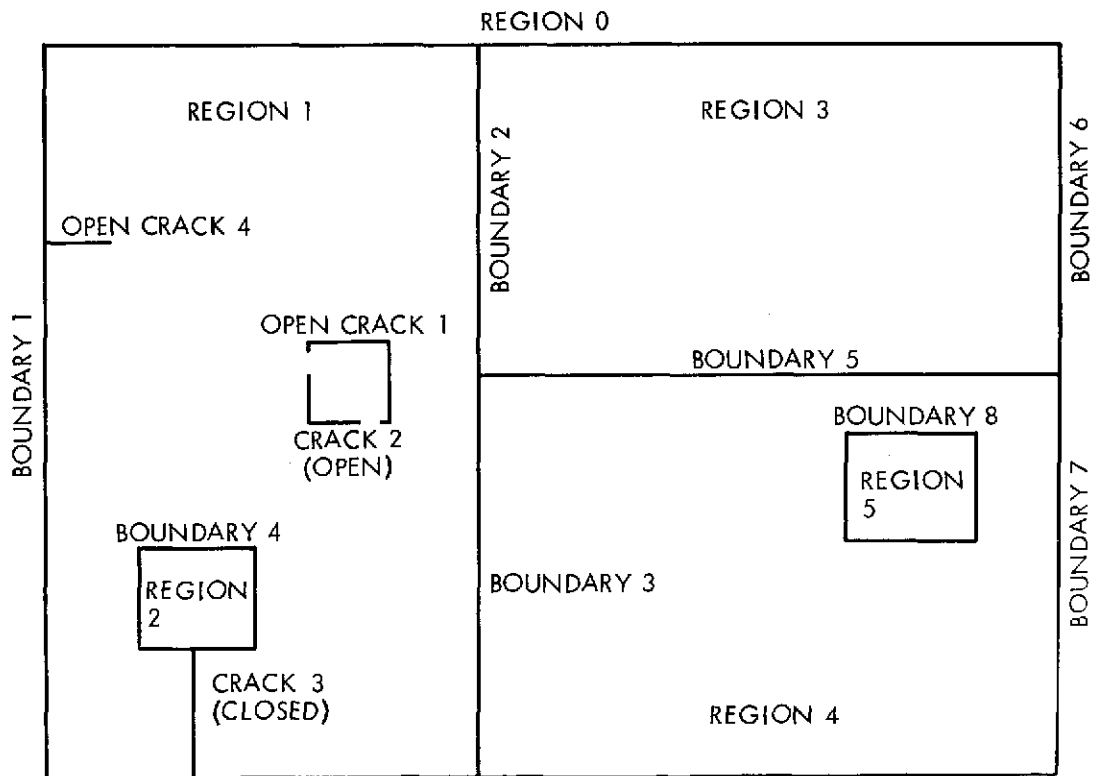
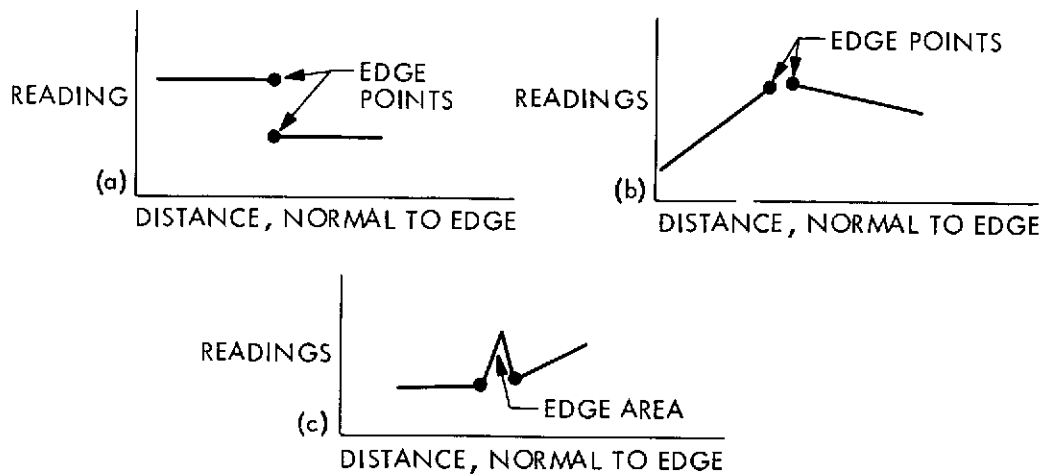


Fig. 2. Illustration of terms



- (a) IDEALIZED STEP EDGE (DOMINANT EDGE TYPE IN VISUAL IMAGES).
 (b) PURE GRADIENT EDGE (CORNERS ARE ESPECIALLY FREQUENT IN ANALYSIS OF 3-D IMAGES WHEN DIRECT MEASURE OF DISTANCE IS AVAILABLE)
 (c) SPIKE EDGE (APPEARS FREQUENTLY IN CORNER EDGES IN VISUAL IMAGES).

Fig. 3. Typical edges

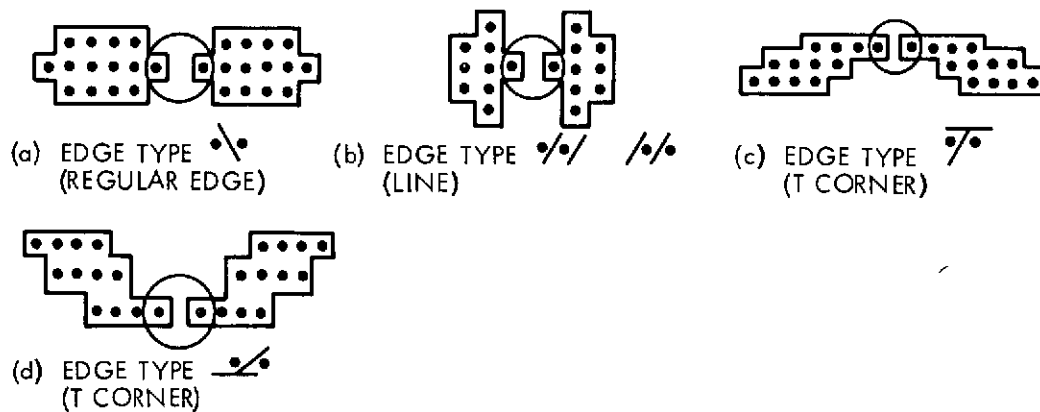
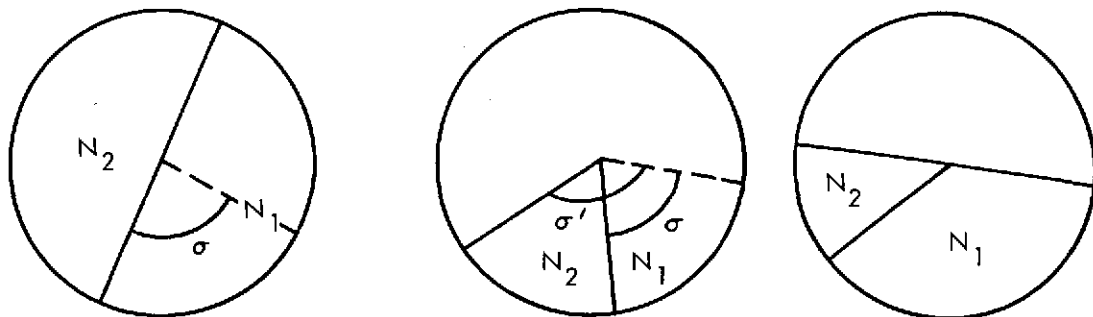


Fig. 4. Typical neighborhoods for edge detection



GENERAL EDGE ORIENTATION
DETECTOR IN DISK COM-
PUTED FOR FINDING OPTIMAL
ORIENTATIONS, DEPENDING
ON THE DESIRED ANGULAR
RESOLUTION

CORNER DETECTOR;
 σ' AND σ TAKEN
TO BE TYPICAL
ANGULAR RESOLUTIONS

T CORNER DETECTOR

Fig. 5. Extended neighborhoods set

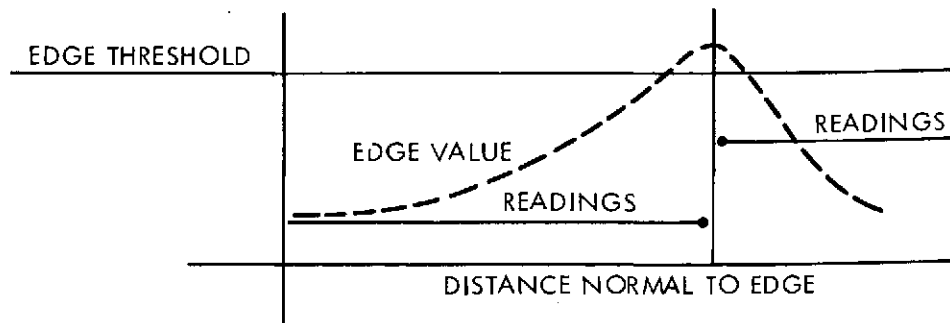
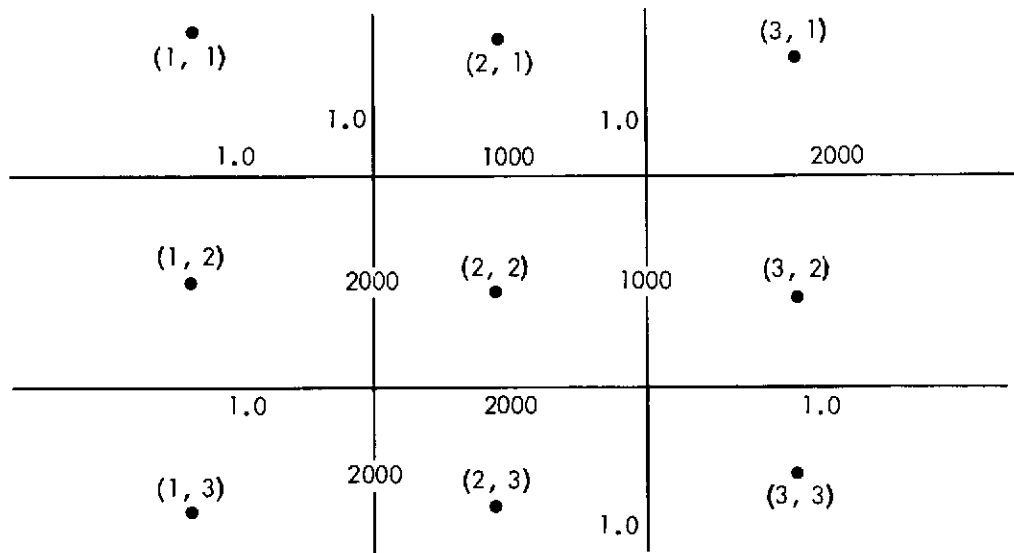


Fig. 6. An ideal edge value cross section



THE (i, j) ARE POINT NUMBERS AND THE VALUES ARE EDGE UNIT VALUES. CLEARLY POINTS $(1, 1)$ $(1, 2)$ $(1, 3)$ $(2, 1)$ $(3, 1)$ SHOULD BE IN ONE REGION AND $(3, 2)$ $(3, 3)$ $(2, 3)$ IN ANOTHER, BUT WHERE $(2, 2)$ SHOULD BE IS TOTALLY AMBIGUOUS (ASSUMING THAT SINGLE POINT REGIONS ARE NOT ALLOWED).

Fig. 7. Region growing ambiguity example

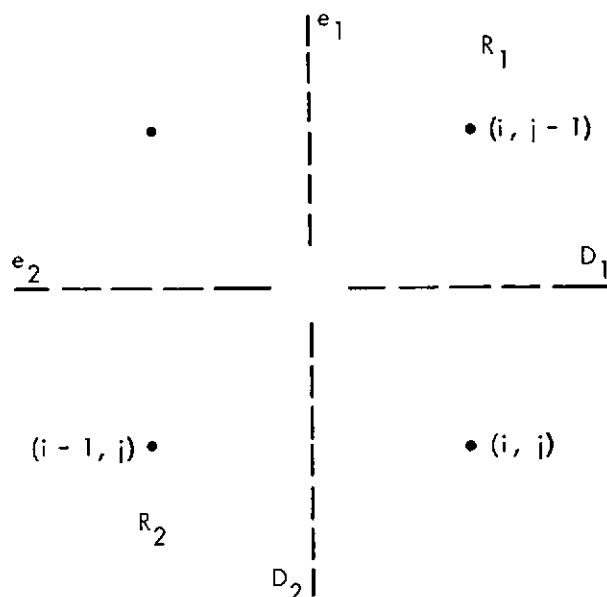


Fig. 8. Algorithm terms definition

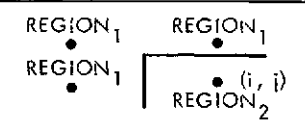
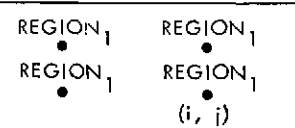
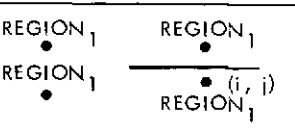
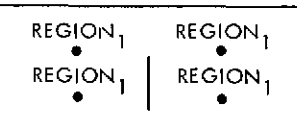
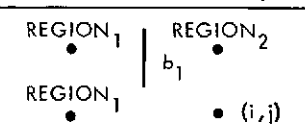
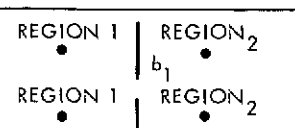
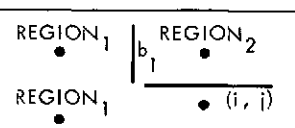
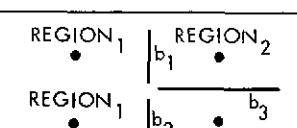
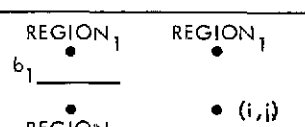
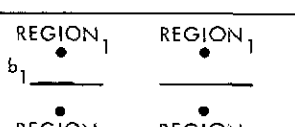
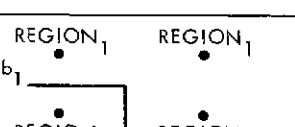
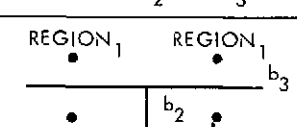
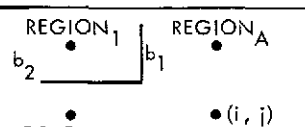
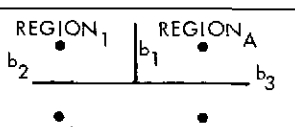
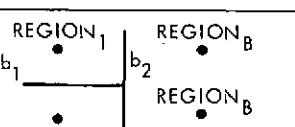
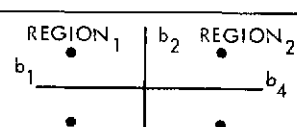
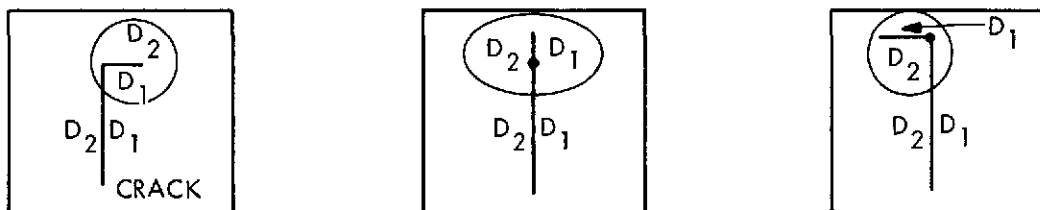
 <p>START NEW REGION (REGION₂). PUT TWO EDGE UNITS STARTING BOUNDARY LINE BETWEEN REGION₁ AND REGION₂</p>	 <p>ADD (i, j) TO REGION₁</p>	 <p>INITIATE AN OPEN CRACK EDGE LINE STARTING BETWEEN (i, j) (i, j - 1) INSIDE REGION₁</p>	 <p>INITIATE AN OPEN CRACK STARTING BETWEEN (i, j) (i - 1, j) INSIDE REGION₁</p>
 <p>MAKE b₁ AN OPEN CRACK. MERGE REGION₁ AND REGION₂ AND ADD (i, j) TO THE UNION</p>	 <p>ADD (i, j) TO REGION₂ AND ADD TO BOUNDARY LINE b₁ ANOTHER EDGE UNIT BETWEEN (i, j) AND (i - 1, j)</p>	 <p>ADD (i, j) TO REGION₁. EXTEND EDGE LINE b₁ BY EDGE UNIT BETWEEN (i, j) (i - 1, j)</p>	 <p>CREATE NEW REGION₃ CONTAINING ONLY (i, j). CREATE TWO BOUNDARY EDGE LINES b₂ AND b₃</p>
 <p>MAKE b₁ AN OPEN CRACK. MERGE REGION₁ AND REGION₂ AND ADD (i, j) TO THE UNION</p>	 <p>ADD (i, j) TO REGION₂ AND ADD TO BOUNDARY LINE b₁ ANOTHER EDGE UNIT BETWEEN (i, j) AND (-1, j)</p>	 <p>ADD (i, j) TO REGION₁. EXTEND EDGE LINE b₁ BY EDGE UNIT BETWEEN (i, j) (i - 1, j)</p>	 <p>CREATE NEW REGION₃ CONTAINING ONLY (i, j). CREATE TWO BOUNDARY EDGE LINES b₂ AND b₃</p>
 <p>IF A ≠ B, THEN MERGE A AND B AND ADD (i, j) TO THE UNIFIED REGION. COMBINE BOUNDARY LINES b₁ AND b₂</p>	 <p>IF A = B, THEN COMBINE BOUNDARY LINES b₁ AND b₂ AND MAKE b₃ A CLOSED CRACK IF A ≠ B b₃ IS CURRENTLY A BOUNDARY</p>	 <p>IF A = B, THEN COMBINE BOUNDARY LINES b₁ AND b₂ AND MAKE b₃ A CLOSED CRACK IF A ≠ B b₃ IS CURRENTLY A BOUNDARY</p>	 <p>START NEW REGION, REGION₄ AND TWO BOUNDARY LINES b₃ AND b₄</p>

Fig. 9. The different region growing decisions



THE 3 OPTIONS TO EXTEND AN OPEN CRACK AND THE
CORRESPONDING ASSUMPTION ON DISTRIBUTING

Fig. 10. The three options to extend an open crack and the
corresponding assumption on distributing

1. Report No. 33-709	2. Government Accession No.	3. Recipient's Catalog No.	
4. Title and Subtitle BOUNDARY AND OBJECT DETECTION IN REAL WORLD IMAGES		5. Report Date November 15, 1974	
		6. Performing Organization Code	
7. Author(s) Yoram Yakimovsky		8. Performing Organization Report No.	
9. Performing Organization Name and Address JET PROPULSION LABORATORY California Institute of Technology 4800 Oak Grove Drive Pasadena, California 91103		10. Work Unit No.	
		11. Contract or Grant No. NAS 7-100	
		13. Type of Report and Period Covered Technical Memorandum	
12. Sponsoring Agency Name and Address NATIONAL AERONAUTICS AND SPACE ADMINISTRATION Washington, D.C. 20546		14. Sponsoring Agency Code	
15. Supplementary Notes			
16. Abstract A solution to the problem of automatic location of objects in digital pictures by computer is presented. A self-scaling local edge detector which can be applied in parallel on a picture is described. Clustering algorithms and boundary following algorithms which are sequential in nature process the edge data to locate images of objects.			
17. Key Words (Selected by Author(s)) Computer Applications and Equipment Information Theory Mathematical Sciences Matter Recognition		18. Distribution Statement Unclassified -- Unlimited	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 28	22. Price

HOW TO FILL OUT THE TECHNICAL REPORT STANDARD TITLE PAGE

Make items 1, 4, 5, 9, 12, and 13 agree with the corresponding information on the report cover. Use all capital letters for title (item 4). Leave items 2, 6, and 14 blank. Complete the remaining items as follows:

3. Recipient's Catalog No. Reserved for use by report recipients.
7. Author(s). Include corresponding information from the report cover. In addition, list the affiliation of an author if it differs from that of the performing organization.
8. Performing Organization Report No. Insert if performing organization wishes to assign this number.
10. Work Unit No. Use the agency-wide code (for example, 923-50-10-06-72), which uniquely identifies the work unit under which the work was authorized. Non-NASA performing organizations will leave this blank.
11. Insert the number of the contract or grant under which the report was prepared.
15. Supplementary Notes. Enter information not included elsewhere but useful, such as: Prepared in cooperation with... Translation of (or by)... Presented at conference of... To be published in...
16. Abstract. Include a brief (not to exceed 200 words) factual summary of the most significant information contained in the report. If possible, the abstract of a classified report should be unclassified. If the report contains a significant bibliography or literature survey, mention it here.
17. Key Words. Insert terms or short phrases selected by the author that identify the principal subjects covered in the report, and that are sufficiently specific and precise to be used for cataloging.
18. Distribution Statement. Enter one of the authorized statements used to denote releasability to the public or a limitation on dissemination for reasons other than security of defense information. Authorized statements are "Unclassified-Unlimited," "U. S. Government and Contractors only," "U. S. Government Agencies only," and "NASA and NASA Contractors only."
19. Security Classification (of report). NOTE: Reports carrying a security classification will require additional markings giving security and downgrading information as specified by the Security Requirements Checklist and the DoD Industrial Security Manual (DoD 5220.22-M).
20. Security Classification (of this page). NOTE: Because this page may be used in preparing announcements, bibliographies, and data banks, it should be unclassified if possible. If a classification is required, indicate separately the classification of the title and the abstract by following these items with either "(U)" for unclassified, or "(C)" or "(S)" as applicable for classified items.
21. No. of Pages. Insert the number of pages.
22. Price. Insert the price set by the Clearinghouse for Federal Scientific and Technical Information or the Government Printing Office, if known.

1. Report No. 33-709	2. Government Accession No.	3. Recipient's Catalog No.	
4. Title and Subtitle BOUNDARY AND OBJECT DETECTION IN REAL WORLD IMAGES		5. Report Date November 15, 1974	
		6. Performing Organization Code	
7. Author(s) Yoram Yakimovsky		8. Performing Organization Report No.	
9. Performing Organization Name and Address JET PROPULSION LABORATORY California Institute of Technology 4800 Oak Grove Drive Pasadena, California 91103		10. Work Unit No.	
		11. Contract or Grant No. NAS 7-100	
12. Sponsoring Agency Name and Address NATIONAL AERONAUTICS AND SPACE ADMINISTRATION Washington, D.C. 20546		13. Type of Report and Period Covered Technical Memorandum	
		14. Sponsoring Agency Code	
15. Supplementary Notes			
16. Abstract A solution to the problem of automatic location of objects in digital pictures by computer is presented. A self-scaling local edge detector which can be applied in parallel on a picture is described. Clustering algorithms and boundary following algorithms which are sequential in nature process the edge data to locate images of objects.			
17. Key Words (Selected by Author(s)) Computer Applications and Equipment Information Theory Mathematical Sciences Matter Recognition		18. Distribution Statement Unclassified -- Unlimited	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 28	22. Price