17694

NAME OF AUTHOR... MELVIN WESLEY SMITH

TITLE OF THESIS... ON THE DETECTION OF EDGES IN PICTURES

UNIVERSITY... UNIVERSITY OF Alberta

DEGREE FOR WHICH THESIS WAS PRESENTED... M. Sc.

YEAR THIS DEGREE GRANTED... 1973

(Signed) M.W. Smith

PERMANENT ADDRESS:

17 Fonseca Street

Aylmer, Quebec

DATED... July 13 ... 1973

THE UNIVERSITY OF ALBERTA

ON THE DETECTION OF EDGES IN PICTURES

BY

© MELVIN WESLEY SMITH

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

AND RESEARCH IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTING SCIENCE

EDMONTON, ALBERTA

FALL, 1973

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled ON THE DETECTION OF EDGES IN PICTURES submitted by Melvin Wesley Smith in partial fulfillment of the requirements for the degree of Master of Science.

Wayne A. Davis
.....
(Supervisor)

L. K. Schubert
.....

Calvert
.....

A. Welch for H. G. Schmidt Warren
.....

DATE... Jan. 17, 1973.

Abstract

This thesis is concerned with the detection of edges in pictures. Edges are defined, the problems of their detection are outlined, and edge operator characteristics are presented. Current detection methods, both sequential and parallel, are critically surveyed. A new edge operator is proposed which is based on the degree of grey level clustering within a picture segment, and two algorithms are included. The results of applying six edge detectors on two digitized pictures are presented. A figure of merit is established for evaluating edge matrices, and the matrices resulting from the six detectors are compared. Hardware edge detection techniques are examined and an approach using hardware is suggested.

ACKNOWLEDGMENTS

The author wishes to express his gratitude to Professor Wayne A. Davis for suggesting edge detection as a topic for investigation in the field of image processing. His comments, suggestions, and criticisms have made the task worthwhile, and his moral and material support have made it possible.

The author also wishes to acknowledge the financial support of the Department of National Defence. Thanks are also due Mr. Norman S. Tsang for his valuable programming assistance. The many discussions with Mr. Rod N. McPherson are also acknowledged.

Table Of Contents

		Page
Chapter 1	INTRODUCTION	1
I.	Problems In Picture Processing	2
II.	Edge Detection	6
III.	Edge Operator Characteristics	14
Chapter 2	SURVEY OF EDGE DETECTION METHODS	18
I.	Sequential Methods	18
II.	Quasi-Parallel Methods	45
III.	Conclusion	55
Chapter 3	ANOTHER EDGE DETECTOR	58
I.	Foundations	58
II.	Region Detection	59
Chapter 4	FUNCTIONAL EVALUATION OF EDGE IMAGES	76
I.	Quality Of Edge Images	76
II.	Line-Producing Algorithms	80
III.	Assessment Of Edge Matrices	84
IV.	Test Environment	86
V.	Test Parameters	94
VI.	Test Results	98
VII.	Conclusions	121

Table Of Contents (Cont.)

Page

Chapter 5 HARDWARE EDGE DETECTION 123

I. Delay Devices 124

II. A Hardware Edge Detection System 127

Chapter 6 GENERAL CONCLUSIONS 137

I. Analytical Evaluation 137

II. Functional Evaluation 139

III. Hardened Edge Detection 143

IV. Closing Remarks 144

Bibliography and References 145

Vita 152

List Of Figures

Figure		Page
1	Types Of Edges	8
2	The Roberts Sub-Matrix	19
3	Vertical And Diagonal Edges	20
4	The Gradient Sub-Matrix	23
5	Idealized Diagonal And Horizontal Edges	23
6	Sample Picture Segments	24
7	Picture Segments For The LaPlacian	28
8	One Dimensional Pixel Array	29
9	The Herskovits 2-D Array	31
10	Vertical And Diagonal Edges For The Herskovits	32
11	An Ideal Horizontal Edge	35
12	Edge Examples For The Statistical Modified LaPlacian	39
13	False Edge Detected By SML	40
14	Balanced Bimodality	61
15	Degree Of Clustering	64
16	Frequency Histogram Of REFPIC	95
17	Frequency Histogram Of FACE	96
18	REFPIC And FACE	101
19	Edge Transforms Of REFPIC	102
20	Edge Transforms Of FACE	103
21	Edge Transforms Of Diagonal	104

List Of Figures (Cont.)

Figure		Page
23	[illegible] Cross	105
24	[illegible] Corner	106
25	[illegible] Circle	107
26	[illegible] Edge Detection System	130
27	[illegible] Picture Matrix	131
28	The Delay Network	132
	Three Edge Detection Modules	134

List Of Tables

Table		Page
1	Fault Table- Complete Image Of REFPIC	108
2	Fault Table- Complete Image Of FACE	108
3	Fault Table- Diagonal (Check Size = 4)	109
4	Fault Table- Cross (Check Size = 4)	109
5	Fault Table- Corner (Check Size = 4)	109
6	Fault Table- Diagonal (Check Size = 8)	110
7	Fault Table- Cross (Check Size = 8)	110
8	Fault Table- Corner (Check Size = 8)	110
9	Fault Table- Circle (Check Size = 4)	111
10	Fault Table- Circle (Check Size = 8)	111
11	Summary Of Evaluation Findings	140

Chapter 1

INTRODUCTION

This thesis is concerned with digital picture processing, a generic term comprising many distinct but related activities. Edge detection is one area of interest whose ramifications are of concern throughout the field of picture processing. The subjective analysis of two dimensional images is a common day-to-day task for many occupations. A doctor examining X-Rays of his patients, a cartographer sifting through air photographs, forensic scientists comparing fingerprints, are a few work-oriented examples of this otherwise ordinary human activity. The advent of the digital computer has stimulated research in the mathematical (and thus more objective) analysis of pictures. With current speeds, storage capacities, and I/O capabilities of digital computers, it is now efficient in cost and time, and becoming more effective in purpose, to assign to the computer some portion of those visual tasks mentioned above. For example, Andrews et al [1] show that an X-ray of a person with Black Lung disease is strikingly identifiable as such, but only with proper computer processing of the original image. The purpose of this thesis is to critically examine contributions to one

specific problem area of computer image processing: the location of edges within pictures. Following a survey another edge detector is introduced which is based on the degree of 'order' within a picture segment. Tests are then conducted on several edge detectors, and conclusions are drawn. A hardware edge detection system is outlined which is computer controllable. The thesis concludes with an overall analysis of all of the above topics.

I. Problems In Picture Processing

The term 'picture processing', in a scientific sense, implies the computer analysis of two dimensional images representing real scenes (as opposed to those operations on arbitrary matrices not derived from pictures). The totality of picture processing tasks and operations, though extensive, can be categorized. A classification where the computer is becoming particularly valuable is in 'pre-processing' of pictures. This thesis will present a thorough examination of one aspect of pre-processing: namely, the detection of the edges of objects within pictures. To this writer's knowledge such a critical survey has not been attempted before, although Griffith [13] does make a contribution to this end. This survey and the associated test results should aid anyone who must choose among several edge detection methods.

Definition Of Terms

A picture is defined as a real-valued spatial function of two variables f(x,y) where, without loss of generality, one can assume it to be of some standard shape and size. The tests (described in Chapter 3) will be concerned with square pictures represented by integer-valued matrices whose array elements are positive, and where the number of rows or columns will not exceed 256. Each element of a picture matrix represents the average value of the brightness over the corresponding region of the original image. Commonly, this brightness value is referred to as a grey level. The elements of the matrices are referred to as picture elements (pixels or pels). The range of values that a pixel can take is called the grey level range.

Notation And Format

The following mathematical notation has been implemented:

- `*` - the multiply operation.
- `**` - the exponentiation operation.
- `| x |` - the absolute value of x.
- `d` - is used in 'dx' to indicate partial differentiation.

Other mathematical symbols take on their customary connotation, unless otherwise specified.

Problem Areas

Automated visual scene analysis is one specific picture processing task. Visual input systems, such as the robot projects at MIT [30] and Stanford [28], the hand-eye project [10], chromosome counting by Ledley et al [24], typically utilize a pre-processor to condition their picture prior to its passage to higher level systems for analysis. Therefore, between the input device, such as a TV camera, and the pattern recognition routines within a system, the video signal must be transformed into a properly organized data structure. This transformation normally concerns the following processes:

a. Digitization

b. Noise filtering

c. Edge detection

d. Contour following

e. Synthesis of line drawing

Digitization is the process of transforming the brightness function of a continuous image into an array of discrete grey levels. This is accomplished through the integration of the image brightness over some small region (sampling), then assigning a grey level to represent this sampled value (quantization). Hardware to perform this task includes an input sensor (e.g., TV camera, flying spot scanner, or perhaps an image dissector), an analog-to-digital (A/D) converter, and controlling modules

to sample and quantize properly with respect to time and position.

Noise filtering concerns the separation of a 'true' grey level from the derived one. There are both hardware and software techniques for approaching this ideal. In the usual case, additive uncorrelated Gaussian noise is assumed. This degradation is sometimes referred to as 'salt and pepper' noise, and can be restrained by low-pass filters implemented in either hardware or software.

Edge detection is the location and marking of all borders between two contiguous regions in the picture matrix. Since this is the theme of this thesis, a more complete (though still intuitive) definition will be given in the next section.

Contour following is the process of operating on the border markers (linking, eliminating, or adding to them) to the extent that the resulting structure is suitably organized for correct abstraction of its features, i.e., physical edges.

The synthesis of a line drawing is, in essence, the transformation of the contour representation of its edges into a compact mathematical form, where the lines (edges) abstracted from the border markers are retained as coordinate end points, as polynomial coefficients, as vector direction differentials, or some other suitable form.

II. Edge Detection

The purpose of any edge detection scheme is to output an indication of the position of borders, or boundaries, of the input scene. The major objectives in the attainment of this purpose are:

a. The input image must be available and appropriately structured.

b. The edge detection scheme should be appropriate for the class of input images.

c. The output data must be suitably organized, and should indicate in some predetermined fashion the position of all borders in the input scene.

The concept of 'edge' is subject to many interpretations (probably because it is primitive, and thus difficult to define). The following indications of what is meant by an edge element and edge detection should suffice.

Intuitively, an edge element (edgel) is a picture element (pixel) on the border, or common boundary, between two neighbouring picture regions which differ in some measureable characteristic by a threshold T.

Edge detection is the process of assignment to each element p(i,j) of a picture matrix P, an indication of the likelihood that the point p(i,j) is an edgel.

Types Of Edges

Consolidating the opinions of Griffith [13] and Herskovits [16], there appear to be four distinct types of edges. Their common underlying characteristic is some form of brightness transition between the picture regions they divide. These edges are shown in cross-section form in Fig. 1.

The slopes on either side of the roof-type edge are not necessarily identical. The ski-jump edge is from Griffith [13], the remainder from both authors. Griffith [13] distinguishes more types than are shown here, but they are not sufficiently different for inclusion.

Texture edges (i.e., those edges where there is no average brightness transition) are excluded from consideration since the operators surveyed here are ordinarily non-effective with this type of boundary. However, Rosenfeld [41] has developed techniques which allow the Roberts operator [36] to perform effectively in a textured environment.

Methods Of Edge Detection

Edge detection methods can be classified under two broad headings: sequential and parallel.

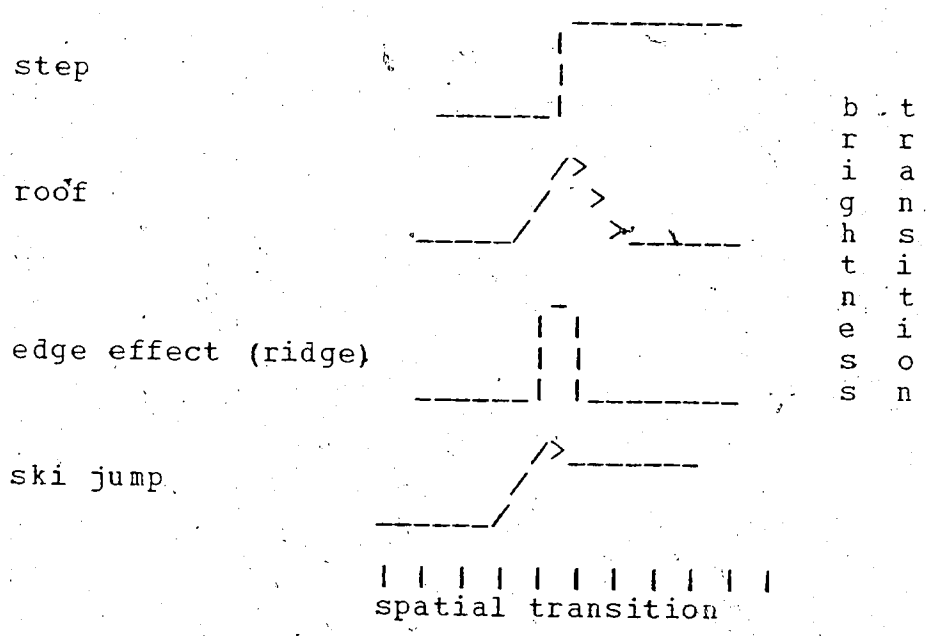


Fig. 1 Types of edges (cross-section)

Sequential

Sequential edge detection methods access and process elements of the input image in some regular serial order, detecting and outputting some form of edge indication sequentially. This type of operation is denoted as local edge detection, and implies that the detection of an edge at element p(i,j) can be made a function of the results from the previous elements accessed (i.e., nonlinear operations are possible). With sequential methods, it is customary to base detection criteria on the characteristics of the neighbourhood around p(i,j). The size and shape of this

neighbourhood (e.g., 3x3, 4x4, etc.) is a parameter of the detection method used.

Parallel

Parallel edge detection methods access and process all elements of the input image simultaneously, producing all the output edge indications at once, i.e., global edge detection. This implies that detection should be much faster than the sequential approach, but that the detection criteria are fixed during any one operation, and are usually identical for all p(i,j).

Implementation Approaches

All of the schemes under consideration in this thesis use optical or computational methods as their prime resource for detection. Electronic methods of edge detection, e.g., using two-dimensional electronic filters and gradient takers [12,27,47] to process a video signal s(x,y), have not been emphasized in the literature in recent years. It is possible that the one dimension of a TV signal causes design and cost problems that severely limit the expansion of 2-D systems for edge detection. However, the advent of charge-coupled semi-conductor devices (CCDs), as reported by Boyle and Smith [2] and others [8], with their capability for analog delay at low cost, and with high performance [46], necessitates a re-examination of these problems. Since their use is currently very restricted,

electronic methods will not be critically surveyed. Instead, a new method of hardware edge detection will be outlined in Chapter 5.

Optical Edge Detection

It has been shown, Preston [35], that when a coherent beam of light (i.e., LASER radiation) is shone through a transparency and an appropriate lens, the Fourier transform of the transmittance of the transparency is produced at the focal plane of the lens. That is, the resulting irradiance pattern is a function of the spatial frequency of the transparency. Higher spatial frequencies show up as regions of high irradiance at distances further from the origin of the transform. These higher spatial frequencies correspond to sharp changes in brightness, i.e., edges. The direction from the origin of these regions is directly related to the spatial orientation of these frequencies. Thus we have implicit edge information embedded in the Fourier transform of a transparency. This information can be extracted by filtering out the low spatial frequencies. The remainder of the transform can then be reconstituted into a real image by passing it through an identical lens placed an additional focal length along the optical axis. This new image will have intensity ridges at locations where the original transparency had edges. The filtering action can be

accomplished by placing an optical stop on the optical axis, thus effectively cutting off all the spatial frequencies that are near zero.

In summary, the above process does a Fourier transformation on the transparency, screens out the near zero frequency components, then re-transforms the clipped Fourier transform into an exo-skeletal image. This image can then be considered as the output of an optical edge detector. The image is not yet in digestible form for edge analysis or recognition routines, but could easily be made so. Through the use of optical methods, edges can be detected over the entire scene at once, i.e., in parallel. Other operations, such as the LaPlacian and the Hadamard Transformation, can also be implemented optically. Both the Fourier Transform and the Hadamard Transform [34] can be implemented as computer algorithms for contrast enhancement, but neither will be discussed further.

Will and Pennington [48] discuss a novel method for the pre-processing of images which attempts to refute the classical methods. Instead of using the natural light from a scene, they illuminate it with a projector shining through a 'crossed grating'. With the resulting gridded image they implement their grid-coding technique to emphasize faint edges on 3-D bodies. They then define edges as the intersection of geometric planes rather than as points of photometric discontinuity.

Computational Edge Detection

Because of the vast amount of information in a typical scene (e.g., a 256 x 256 picture matrix), and the consequent processing required, computational edge detection implies the use of a computer. Associated with the computer are peripheral devices which sense the input image, quantize it, then store the quantized representation (disk, tapes, storage scopes), and which can display various representations of the image (TV monitors, storage scopes, graphic displays, etc).

Edge detection is accomplished through the use of some edge detecting algorithm implemented as a computer program. The program operates on the stored quantized image, and either replaces the image with its edge transform or creates a new structure containing the edge information. This final structure is then normally used as direct input to edge analysis or pattern recognition routines.

Hardware Versus Software

Although hardware of some sort must be used in all methods of edge detection, software edge detection is distinguished as that class of methods where the hardware used only supports the implementation of a computer program. With this in mind, hardware edge detection can be compared to that of software under four criteria: flexibility, speed, accuracy, and cost.

Flexibility: Using a parallel approach essentially limits optical systems to linear edge detection operations on the input image. On the other hand, the computer can perform non-linear operations, and can change detection criteria during any one operation.

Speed: The speed of the edge detection operation for hardware systems is limited only by the speed at which the elements can be accessed. For an optical system, only the speed of light limits the speed with which edge detection is accomplished. Thus, detection is essentially immediate.

Accuracy: With software methods there are inherent errors due to sampling and quantization operations which degrade to some arbitrary extent the output from the sensor. However, even this output has been previously degraded by the imperfect accuracy of the sensor itself. For hardware systems, both electronic and optical, accuracy is a function of the quality of the transmission components.

Cost: Although one can certainly find counter examples, it is generally true that digital computer edge detection is more expensive than optical or electronic means. However, the computer can be used in more than just its edge detection role, and thus its cost can be distributed over other tasks.

Summarizing, it is felt that the main advantage of hardware edge detection is speed and cost, where the advantage of the software method is flexibility, and perhaps accuracy.

Most of the remainder of this thesis will concern sequential computational methods because of their flexibility, interest, and widespread use [26].

III. Edge Operator Characteristics

In Chapter 3 several edge detectors are analytically evaluated. This evaluation can be handled best by grading their several features against a selected set of characteristics. An additional active evaluation (i.e., implementation and controlled trial) of their working performance could then confirm or deny some of the analytical conclusions, and perhaps give evidence about characteristics where analysis is difficult. It follows therefore that the assessment of a method depends upon the selection and weighting of the arbitrary characteristics, and a proper active evaluation. According to Roberts [36], edge detection operators should produce sharp edges with as little background noise as possible. It was also stated that the intensity of the lines produced should correspond closely to a human's ability to perceive the edge in the original picture. This last point is debatable. A human can 'perceive' an edge even though that edge is not clearly visible (i.e., one creates an edge in order to form the

object recognized). If such is the intent of that last point, then there is no debate. However, if an edge operator produces a faint edge (or none at all) when a human can definitely perceive one, then the production of this faint edge is not a characteristic of a good operator. In fact, the 'perfect' operator should produce a strong edge response even though the human viewer has difficulty doing so. Bearing these points in mind, the following composite list of characteristics is clearly relevant to the selection and use of an edge detecting method:

a. Speed: the amount of time consumed to determine the likelihood that one pixel is an edge element. As an arbitrary measure, a Computer Time Unit (CTU) will be used, where it is assumed that the ADD instruction operations and the storage access time are both approximately equal to 1 CTU and thus about equal to each other. Further, it is assumed that the MULTIPLY and DIVIDE times are completed in about 4 and 8 CTUs respectively. COMPARISON operations should consume approximately 2 CTUs, while an ABS value operation should use about 4 CTUs. MAX or MIN operations should take 3 CTUs (i.e., a subtraction and a comparison) since the operands will probably be in registers when needed. Henceforth, these calculated times for each operator should not be interpreted as exact times, only as approximations.

b. Isotropy: the insensitivity of an operator to changes in orientation of edges. However, a isotropy is certainly desireable if one wishes to detect edges oriented in a specific direction.

c. Edge Type Response: the response of the operator to different types of edges.

d. Dynamic Range: effectiveness of the operator over wide ranges of brightness.

e. Noise Sensitivity: ability to indicate correct edge elements under varied noise conditions.

f. Storage Costs: economical use of core and auxiliary storage.

g. Universality: degree of machine/system independence of the operator.

h. Adaptability: ability to change, expand, or otherwise adapt the parameters of the operator to changing picture conditions, while the operator is actively engaged in processing a picture.

The weighting of these characteristics is difficult, since the individual user must decide on the relative importance of two (or more) features, which may be counter-acting (e.g., adaptability as opposed to fast response). However, the functional effectiveness of any method can be assessed by examination of its output, when given a 'real' picture as input. This assessment is somewhat subjective however, and therefore will vary under different checking conditions. A later chapter will discuss a more

objective basis for judging the merit of any particular edge operator under a wide range of operating conditions.

For the present analytical evaluation, the characteristics of speed, isotropy, and edge type response will be referred to during discussion of each method. Other features will be mentioned when clearly relevant.

Performing edge detection on a visual scene through computational methods implies that the scene be properly represented in mathematical form. Normally this representation is a 2-dimensional array of non-negative integers whose range is $0 \leq n < 2^{**}k$, where the exponent k usually lies in the range $0 < k \leq 6$. The size of the array should be such that the smallest resolvable feature of the original picture is much larger than the individual pixel.

Chapter 2

SURVEY OF EDGE DETECTION METHODS

This chapter will examine twelve methods of edge detection. The order of presentation is related to their relative complexity, with the simple sequential classical methods being presented first, followed by the more modern sequential operators. These sequential approaches are terminated with a review of statistical differencing algorithms. Finally, quasi-parallel methods are reviewed in lesser detail. All methods discussed are either recent or widely used.

I. Sequential Methods

This section critically surveys several sequential methods of local edge detection in preparation for functional tests of a selected subset of them in chapter 4. The methods discussed include: the Roberts operator, the Gradient and LaPlacian operators, along with the newer Herskovits operator. Methods involving statistical operations are discussed separately. For simplicity, analysis of the following edge detectors in this section is, in general, restricted to their response to step-like edges.

The Roberts Operator

The Roberts operator [36,1] was designed to detect gradients of arbitrary orientation but of minimum width. That is, an attempt is made to locate micro-edges in all directions. Using a 2 x 2 pixel window (shown in Fig. 2),

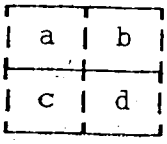


Fig. 2. The Roberts sub-matrix

the whole picture matrix was examined with the following simple algorithm:

$$\text{Edge Merit (EM)} = \sqrt{(A-D)^2 + (C-B)^2} \quad (1)$$

where A,B,C,D are the square roots of the corresponding grey level intensities (i.e., a,b,c,d) in the picture. This initial root operation is used since it approximates the response of the human eye to increasing values of brightness [14]. Perhaps a somewhat more useful explanation would be that the root operation effectively normalizes the dynamic range of grey levels, such that the threshold needs minimum adjustment for effective operation. However, it is common (see Rosenfeld's work on texture

analysis [39], for example) to omit one or both root operations. For the present assessment the pre-processing root operation will be omitted for reasons of efficiency.

This algorithm emphasizes differences across the window, both horizontally and vertically. These differences are indicative of edges. Both terms under the square root sign contribute to the EM of the vertical edge shown in Fig. 3a.

a. Vertical b. Diagonal

Fig. 3. Vertical and diagonal edges

It is clear that a horizontal edge would receive a similar contribution from both terms; however, the diagonal edge in Fig. 3b, although apparently as strong, would receive only about two-thirds of the EM received by an equally strong horizontal or vertical edge.

With the assumption that the pixels represent the actual brightness intensities in the picture rather than their roots, the speed of the operator is still not outstanding: a calculation shows four storage accesses, two

subtractions, two multiplications, one addition, one square root operation (approx. 50 CTUs), and one comparison operation, for a total of about 67 CTUs. This total is largely comprised of just one component, the square root operation. This operation can be omitted (to optimize the algorithm) under the assumption that the comparison operation will be done against a squared threshold. With this change an operator time of approximately 17 CTUs results, which is indeed much faster.

The storage requirements of the operator are clearly minimal since only 17 CTUs are used, indicating few instructions. The major extra storage requirement is the necessity to maintain an edge indicator matrix whose dimensions are the same as the picture. The fact that the examined region is small, (2 x 2), and is analyzed out of context with neighbouring regions, hinders correct operation in the presence of noise, both high frequency and 'salt and pepper'. As noted above, Rosenfeld [41] used a modified version of this erator. Compared against three other algorithms, Rosenfeld rated this method second best in the attempt to establish texture boundaries. The Roberts operator does this by detecting the amount of micro-edge per unit area. This conclusion is developed and tested in [41].

The Gradient

For most purposes one wishes to detect edges at every orientation over the picture matrix. Note that the Roberts operator is weak on diagonal edges because of the size of the examined area. Now consider a method which supposedly indicates edges equally well in any direction, but through expedient use, is more responsive to diagonally oriented edges as opposed to vertical or horizontal ones. This method calculates the derivative in the Gradient direction [38,pp.94-95]. This amounts to taking the square root of the sum of the squares of the partial derivatives in any pair of orthogonal directions:

$$[(df/dx)^2 + (df/dy)^2]^{**}.5.$$

Falk [9] uses this method in a complex scheme to detect edges from imperfect (i.e., noisy or missing) images. With this scheme it is customary to use a 3 x 3 subset of the picture as the region of interest. The individual pixels will be referred to hereafter by the letters shown in Fig. 4.

Various ways are suggested for implementing the Gradient, but a typical and symmetrical variation is to sum the positive differences of grey levels across the central pixel such that we have the following digital approximation:

$$EM = |b-h| + |f-d|. \qquad (2)$$

It is to be noted that pixel e does not enter into the calculation, even though it is the element about which the above equation speaks. Further, note that the four corner elements of the region are also missing from the calculation. Apparently, their absence aids the speed of the

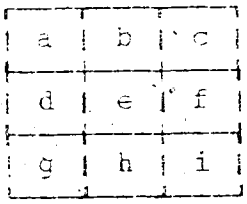
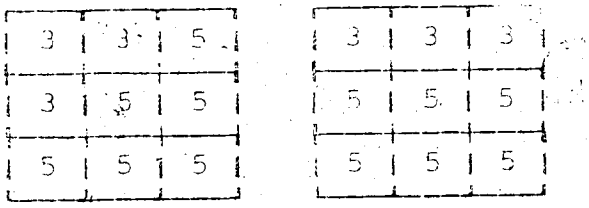


Fig. 4. The Gradient sub-matrix

calculation much more than their inclusion would aid a correct response from the operator. As an example of the operator's action consider the two 3 x 3 picture segments in Fig. 5, which include part of an idealized diagonal edge.

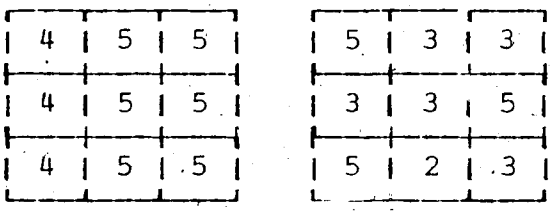


a. Diagonal b. Horizontal

Fig. 5. Idealized diagonal and horizontal edges.

Using equation (2) above to calculate EM, a value of 4 is derived. Thresholding at a level of 2 would result in a

strong edge indication. Consider now the horizontal edge in Fig 5b. Note now that EM = 2, just barely indicative of an edge. With such a marginal threshold, the operator can give false edge indications. Using a threshold of 2, examine the two segments of Fig. 6. As can be calculated, Fig. 6a gives



a. Obvious Edge b. N Edge

Fig. 6. Sample picture segments

no indication of an edge, whereas Fig. 6b gives a definite indication. Since the reverse situation is the correct one, the response of the operator using this threshold is unsatisfactory. A large part of this difficulty is due to the inequitable response to edges that are 45 degrees apart. Therefore, isotropy is only closely approximated when the definition of the Gradient is expediently implemented.

Rosenfeld [38] also suggests the following Gradient operator:

$$EM = |(a+b+c)-(g+h+i)| + |(a+d+g)-(c+f+i)| \qquad (3)$$

This operator attempts to overcome salt and pepper noise by examining more of the pixels of the picture segment during one edge operation. Where the additional CPU time needed is

of no real consequence, this variation may be worthwhile. However, again note the disparity between vertical and horizontal edges versus diagonal ones. Using Fig. 5 again, it is seen that EM = 8 for the diagonal edge (Fig. 5a) and 6 for the horizontal edge (Fig. 5b). Thus, unequal response to edges at different orientations, from an operator whose purpose is edge detection at all orientations, leaves something to be desired. An alleviation to this problem was suggested by Holmes [17]:

$$\text{Let } dx = ((c+f+i)/3) - ((a+d+g)/3)$$

$$\text{and } dy = ((a+b+c)/3) - ((g+h+i)/3)$$

$$\text{then, } |\text{grad}| = (\text{MAX}(|dx|,|dy|)) + (3/8)*(\text{MIN}(|dx|,|dy|)). \quad (4)$$

This version almost attains isotropy, but at the expense of about 76 CTUs.

A time analysis of (2) shows that 17 CTUs are required, which is identical to the Roberts operator. The second variation, i.e., (3), of the Gradient uses eight extra memory accesses and additions for an extra 16 CTUs. Thus, its total of 33 CTUs is approximately double the first variation. No version of this operator has excessive storage requirements over the necessary edge matrix required.

Since difficulty was encountered in obtaining equal response from edges that were 45 degrees apart from the Roberts Operator and two versions of the Gradient, and since

the anisotropic effects were opposite in sign, a combination of both of the operators should give near isotropic results. A suggested variation is the following:

$$EM = |a+d-i-f| + |g+h-b-c|. \qquad (5)$$

It can be confirmed that the EM of Fig. 5 using this combination of operators is 6 for both diagonal and horizontal edges, and uses less CPU time than the second variation of the Gradient. Note that 8 of the 9 pixels in the segment are used in the calculation, with better results in the presence of noise. This is re-confirmed by the same computation on Fig. 6b.

The Laplacian

The LaPlacian is an approach which is oriented toward the detection of thin high contrast edges (e.g., highlights). Kovasnay and Joseph [23] were the first to discuss this operator. Rosenfeld [38,p.95] mentions the operator as "another useful combination of derivatives ...". For a continuum, the Laplacian is defined by:

$$d^2f/dx^2 + d^2f/dy^2.$$

A digital approximation of this function using the lettering scheme from Fig. 4, is:

$$EM = 4e - (b+h+f+d). \qquad (6)$$

A second, more noise-free, variation [38] is:

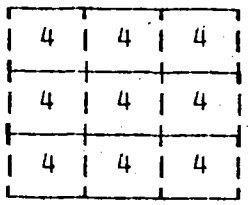
$$EM = 8e - (a+b+c+d+f+g+h+i). \qquad (7)$$

Both of these expressions tend to maximize in regions where there is an abrupt change of slope. Since the most abrupt change is from strongly positive to strongly negative, it can be seen that this method is more responsive to a thin edge (i.e., one that can be straddled by the operator) than to a wider edge.

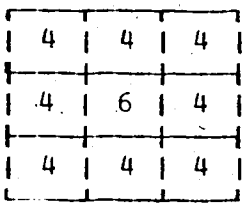
Although it is clear that the Laplacian is intended to be isotropic (as have previous methods), one notes anisotropic results using equation (6) on the segments of Fig. 5. That is, for the diagonal edge, EM=4, and for the horizontal edge, EM=2. Thus, the first variation is more positive for diagonal edges (as was the Gradient). Applying the second variation (7) to the segments, one notes that EM is the same in both vertical (horizontal) and diagonal edges. Therefore, the second variation produces a more isotropic detector output. Also, one can expect the effects of noise to be lessened because of the greater number of pixels examined.

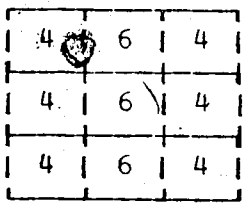
The speed of the operator using (6) is 15 CTUs, and is 23 CTUs for (7). As with previous methods, the storage requirements are minimal.

The most serious objection to this method is its susceptability to salt and pepper noise. Using (7), consider the segment in Fig. 7a. Obviously, the central pixel is not an edge element. Yet, the addition of a small amount of noise to it (see Fig. 7b) results in EM=16, whereas a



a. No edge





b. Noise point c. Thin vertical edge

Fig. 7. Picture segments for the LaPlacian

definite edge (see Fig. 7c) has EM=12. This result was obtained from the variation which would seem to be highly noise-resistant (7). As the reader can easily verify, variation (7) is the addition of two LaPlacian operations: variation (6) plus the same variation, but using instead the pixels a,c,g,i in the subtracted term. Since the LaPlacian is basically a 'spike' detector, the noise point (spike) in Fig. 7b is more strongly detected.

Thus if the picture has minimal noise, the LaPlacian would be productive for thin high contrast boundaries in the scene.

The Herskovits Operator

Herskovits [16] developed a one-dimensional operator based on the geometric appearance of an ideal edge. This operator is highly positive to edges which are step-like in appearance. Thus in Fig. 8 below, curve x would be selected before curves y and z, since the latter two curves are not singly-stepped over the operator's range of interest.

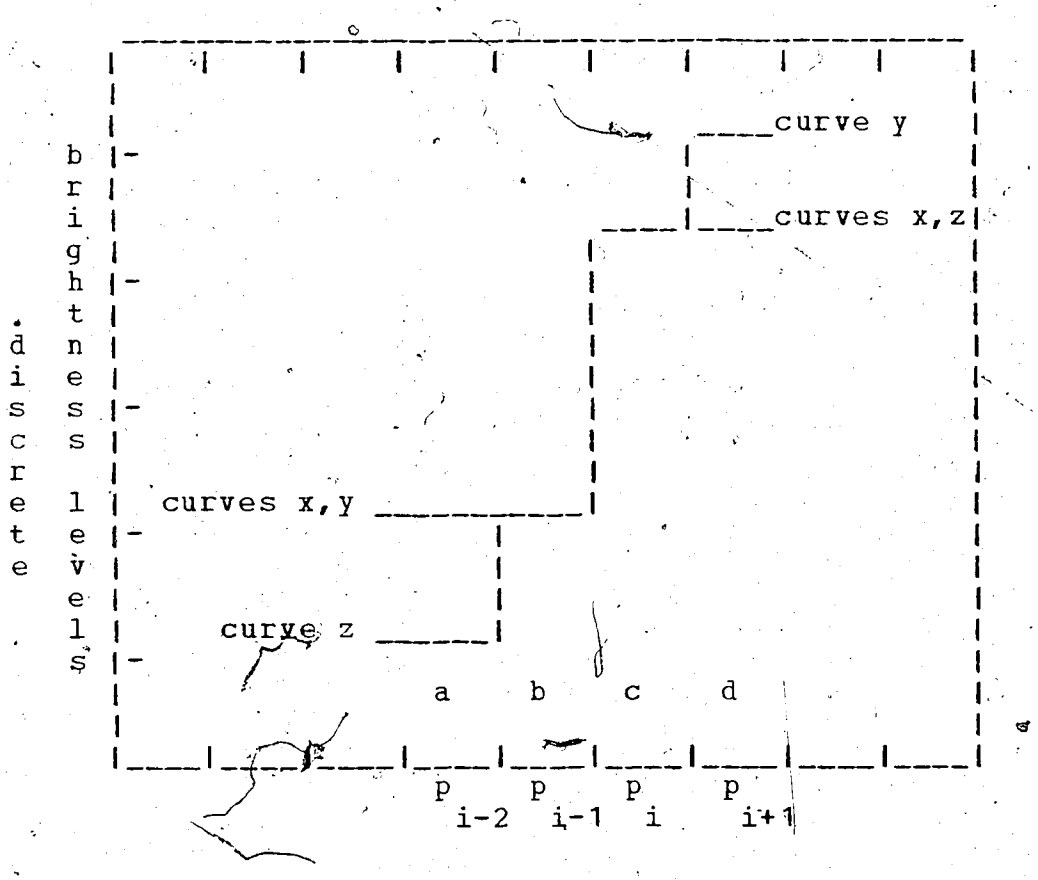


Fig. 8. One dimensional pixel array

The operator involves four consecutive intensity values, a,b,c,d, obtained at equal intervals along a line normal to the orientation of an expected edge. See, for

example, the four marked intervals of p in Fig. 8. The formula is:

$$EM = 2(b-c) - (a-b) - (c-d). \qquad (8)$$

The first term on the right-hand side of (8) expresses difference in intensity as edge-like. The second two terms argue for flatness before and after the edge respectively.

The above equation, as reported by Griffith [13] and originating in [16], leads to two intrinsic difficulties. First, different EMs are obtained for mirror image edges; second, the EM depends on the sign of perturbations in front of, or behind, an edge. Both difficulties can be overcome by taking absolute values of each term in the equation. With these changes the following equation results:

$$EM = |2(b-c)| - |a-b| - |c-d|. \qquad (9)$$

Previously discussed edge detectors (i.e., the Roberts, Gradient, and the LaPlacian) have been designed with isotropy in mind. To expand the Herskovits operator for omni-directional edge detection it is only necessary to apply it on a second scan over the scene in an orthogonal direction to that of the first scan. Combining (i.e., adding; or better still, squaring, adding, and rooting) the results of both scans will produce an EM which approximates isotropy. This EM can then be arbitrarily thresholded for an edge indication.

To enable easy comparison with other approximately isotropic methods, it is suggested that the following 2-D operator be used:

$$EM = MAX(|2(b-c)| - |a-b| - |c-d|, \; |2(f-c)| - |e-f| - |c-g|). \quad (10)$$

An examination of a picture segment lettering scheme given in Fig. 9 shows that the operator (10) is capable of some

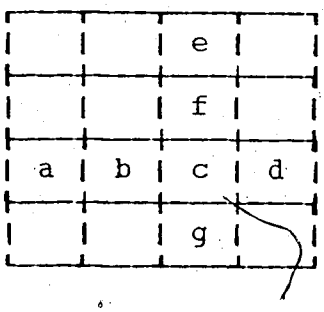
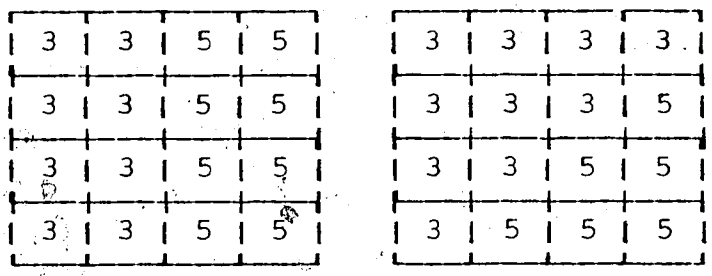


Fig. 9. The Herskovits 2-D array

degree of isotropy. Further, it seems fair to check that degree even though the extended version of the operator has been formed within this thesis, since the modification seems the most reasonable in terms of speed and complexity. Using the sum of horizontal and vertical edge merits (rather than the MAX operation), though seemingly reasonable, discriminates against certain diagonal edges, and so was not seriously considered.

Consider now the picture segments in Fig. 10. The modified Herskovits Operator gives an EM = 2 for the vertical edge, and an EM = 2 for a diagonal edge. Thus, in this form, equation (10) is isotropic for the special cases shown.



a. Vertical b. Diagonal

Fig. 10. Vertical and diagonal edges for the Herskovits

In its modified form (10), the operator uses eight memory accesses, two multiplications, ten subtractions, one addition, one comparison, six ABS value operations, and one MAX operation for a total of 59 CTUs. This is considerably slower than previously discussed local methods.

With this method also, it is necessary to maintain a binary matrix in correspondence with the picture matrix to contain the edge indicators. The storage required by the operator itself, however, is fairly small.

The Shirai-Tsuji Operator

The last of the local sequential operators to be considered has been arbitrarily named the Shirai-Tsuji Operator for its authors [45].

This method can be considered as a variation on Directional Differencing (mentioned by Rosenfeld [38,p.100] as a further variant on the Gradient method). The operator computes the horizontal difference Dx and vertical difference Dy across a 3 x 3 picture segment, then combines Dx and Dy into a semi-isotropic value. A further operation on Dx and Dy derives the direction of the gradient vector K. The operator is formally defined [45] as:

$$Dx = ((a+d+g) - (c+f+i)) \quad (11)$$

$$Dy = ((a+b+c) - (g+h+i)) \quad (12)$$

$$D' = (Dx+Dy)/3 + Kb \qquad (Kb = \text{constant}) \quad (13)$$

$$Alpha = \arctan(Dy/Dx) \quad (14)$$

$$D = D' \text{ if } D'>0, \text{ or } D = 0 \text{ if } D' \leq 0 \quad (15)$$

$$K = (20/pi)*Alpha \pmod{20} \quad \text{(gradient direction)}. \quad (16)$$

There is an anomaly in this operator which bears closer examination. The value D is an indicator of the likelihood that element e is an edge element. The equation for D indicates that D' might possibly be a negative value. Thus, it is reasonable to assume that either Dx and/or Dy may take on negative values. Therefore, it is possible that Dx=4, and Dy=-4. With these values one notes that D'=0,

indicating no edge, but it is obvious that these values represent a strong diagonal edge, which should be recognized by the operator. To correct this situation, it is suggested that equation (13) for D' be restructured as follows:

$$D' = (|Dx| + |Dy|)/3 + Kb \qquad (Kb = \text{constant}). \qquad (13a)$$

It can be seen here that Kb is a negative valued constant which allows the edge threshold to be set at zero. In addition, one notes that the gradient direction K takes on real values from 0 to 19, where possibly 20 discrete gradient directions were intended. Division by zero in the calculation is also normally handled by ordinary arctan functions.

With these alterations and assumptions one notes that the range of D' straddles zero, with zero being the threshold value. Any change of this threshold is accomplished by changing the bias in the equation for D' rather than changing the threshold itself. The bias value is, of course, Kb.

Now consider the action of this operator on vertical (horizontal) and diagonal edges to determine its effective isotropy. An ideal horizontal edge is provided in Fig. 11.

Note that:

Dx=((13)-(13))=0

Dy=((9)-(15)))=-6

Alpha=arctan(-6/0)= 3(pi/2)

D'=(6/3)+Kb=2+Kb.

Assume that Kb=-1. For the sake of reasonableness, it must

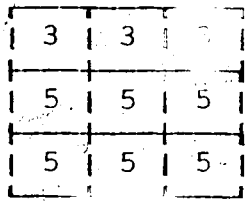


Fig. 11. [illegible] horizontal edge

be set at some negative value [illegible] yet it should allow a relatively strong edge to [illegible] the threshold easily; certainly the present edge is strong. Then we have:

D' = 2+(-1) = 1 and,

K = (20/pi)*Alpha = 10 (mod 20).

An additional computation with an edge of the same strength, but oriented diagonally shows that D'=1.666.

This comparison shows again that isotropy is hampered by addition of horizontal and vertical values. The diagonal edge gives an edge indication which is 66.6% more positive than a horizontal one of equal step height.

As Roberts [36,p.50] has observed, the case for some sort of root operation on the orthogonal values is warranted.

Computing the speed of the operator, note that the calculation of EM (i.e., D') requires ten additions, two subtractions, one division, twelve memory accesses, one absolute value operation, and one comparison for approximately 42 CTUs.

To mechanize the calculation of the gradient direction requires one division (Dy/Dx only, since the remaining division by pi would be stored as a constant), one multiplication, one MOD 10 operation (one division by ten, and one storage access on the remainder), plus the time taken for an arctan operation (typically 11 CTUs). Therefore, the total time for calculation of K is 32 CTUs. Thus a complete operation would consume 74 CTUs. However, the inclusion of gradient K calculations will undoubtedly speed up a later stage in the pattern recognition process. That is, contour following is made more efficient if information is available concerning the probable direction of prospective edges. Thus the extra 32 CTUs are, in fact, a bonus. With 42 CTUs remaining, it is obvious that this method is comparable to the speeds of the other local edge operators.

Because 8 of the 9 pixels in the examined picture segment are used in the edge calculation, this operator is less susceptible to false alarms (i.e., detection of an edge

element where none exists). Indeed, with the derived directional information from the arctan operation, it is possible that a lower threshold can be used (i.e., an increased value of Kb) to detect weaker edges, where other methods would necessarily have to maintain higher thresholds. This operator requires space to store its binary edge matrix, plus the code for the operator itself.

It must also be mentioned here that the Shirai-Tsuji operator sequentially follows the action of their own unique noise filter. The filter selectively and successively adjusts the various grey levels of the 3 x 3 picture segment using heuristics (which are not discussed here) that maintain the resolution of polyhedral shapes while weakening and smearing others. Therefore, the edge detector is designed for a noise-free environment; however, the criticism of anisotropy still holds.

Statistical Differencing

As Rosenfeld [38,p.100] points out, "Rather than differencing by simply subtracting the (average) grey levels in two neighboring regions ...one can compare the FREQUENCY DISTRIBUTIONS of the grey levels in the two regions".

Under the general heading of statistical differencing, several related methods can be found; however, they are sufficiently unique to bear individual attention here. The statistical modified LaPlacian [38] modifies a

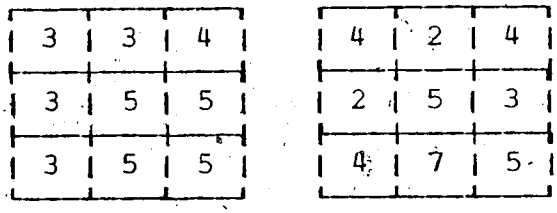
LaPlacian-like operation on a picture segment and includes a standard deviation measure to detect edges. The distributional differencing method examines a histogram of the pixels from two picture regions to determine dissimilarity, and thus gains an indication of edge likelihood. Griffith's method [13] introduces Bayesian probability along with statistical procedures to perform edge detection.

Statistical Modified Laplacian (SML)

It has been suggested [38,p.100] that a useful way to implement the concept of frequency distribution of grey levels would be to assign edge merit to a pixel p(x,y) in direct proportion to the function (f-Av)/s, where f is the value of p(x,y), Av is the average of all p(i,j) in a normally symmetric neighbourhood N around p(x,y), and s(x,y) is the standard deviation over this neighbourhood. This method is arbitrarily named the Statistical Modified LaPlacian because of its characteristics.

This operator can be justified intuitively on the grounds that the function increases in value as the value of a point increases over the average of its neighbouring points (thus indicating a high point, or micro-edge), and is restrained in value if the standard deviation indicates that there are many points surrounding p(x,y) which are far from the average (thus indicating a 'noisy' neighbourhood). This is illustrated by the 3 x 3 picture segments in Fig. 12, and

then compute EM = (f-Av)/s. Note that the standard deviation for Fig. 12a is 1.0, and thus the EM is 1.125. However, for Fig. 12b s=1.59, and EM=.708. Therefore, with the same average value over a neighbourhood, and with the same central pixel value, a higher edge merit indication is obtained from the segment which is subjectively more edge-like.



a. More edge-like b. Less edge-like

Fig. 12. Edge examples for the Statistical Modified LaPlacian

An argument against this approach is illustrated by Fig. 13. It is noted that EM=1.125, but the segment is not impressive as being edge-like. Yet the EM is the same as the one with definite edge-like qualities (i.e., Fig 12a). This ambiguity is due to the fact that computations for standard deviation do not concern themselves with the position of the samples relative to their neighbours, only with the values themselves. However, as just shown, relative positions of the elements are of prime importance in edge indication.

A method which takes into account the relative position of the elements in a picture segment is proposed in Chapter 3.

Assuming a squared threshold, a neighbourhood of 9 elements, and considering that each element must be accessed twice, then a speed of 111 CTUs is estimated. Much of this time is spent computing s**2 (approximately 72 CTUs).

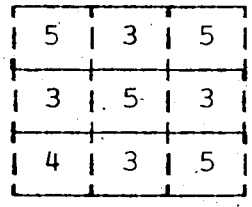


Fig. 13. False edge detected by SML.

Distributional Differencing

Muerle and Allen [31,p.3-15] use a variant of the Kolmogorov-Smirnov two-tailed hypothesis test to indirectly determine edges in scenes.

Briefly, this hypothesis test confirms that two independent sample sets have been drawn from populations having the same statistical distribution. Utilizing this test on the grey levels of a scene should then enable one to determine if two independent (but neighbouring) sets are likely to be part of the same object. If the test strongly rejects this hypothesis then one assumes that the sets are

from different objects, and the pixels on the border between the sets are indicated as edge elements.

To determine the similarity of the two sets, one can construct the cumulative probabilities for both sets, then take the maximum of their positive differences as the EM. If this EM exceeds a prescribed threshold, the sets are from different objects. As Muerle and Allen have determined, this method gives mediocre results, since the maximum difference between the two sets (each containing only one value of grey level, and also differing by only one grey level) may be very large, which is misleading and can indicate gross dissimilarity. In this case, one would probably want to show both sets as part of the same object, but no meaningful threshold would allow this.

Improved results were gained by using "...the average magnitude difference between the two cumulative distributions...rather than the difference at any point ..."[31,p.10]. This average was approximated by determining the area under the positive difference curve for the two distributions.

Great difficulty was experienced [31] with pictures which were composed primarily of edges (e.g., alphabetic characters) since the compared sets of grey levels had no distinct distributional differences (i.e., the backgrounds are the same on both sides of the 'edge' of a character).

Muerle and Allen give no implementable algorithm in their paper, and so a speed check on their operator cannot be easily performed; however, an estimate for one local operation can be made.

With m levels of quantization, assume that one of the sample sets has its cumulative probability values stored, and that a neighbouring set of n elements is to be compared against these values. A rather detailed calculation shows that approximately mn+10(n+1) CTUs are required for n**.5 edge indications (i.e., assuming square neighbourhoods of n elements, one is normally concerned with the 'root of n' elements along the border). For m=16 and n=4, a speed of 114 CTUs results---slow compared to most of the previous methods. The adjustment of the stored values is additional (which prepares for the next local operation), if the regions are found to be from the same object.

The setting of the threshold was another troublesome aspect for the designers [31,p.12] "Further, we feel that the optimal threshold is a function of the mean and standard deviation of the grey levels in the pattern scene to be processed".

Their conclusion concerning threshold requirements is strikingly related to the foundations of the Statistical Modified Laplacian method discussed previously.

An argument which weighs strongly against Distributional Differencing is the same one used against the Statistical Laplacian, i.e., insufficient note is taken of the relative position of grey levels in a set, only their relative values. A picture segment could easily be constructed to illustrate this.

Griffith's Method

A somewhat more complex distributional method will now be briefly discussed. Lesser attention is given to this algorithm because of its less than optimal results on a restricted class of edge types.

Bayesian probability has been used [13] to develop an operator which is positive for narrow highlights, in addition to a class of edges which can be described as cliff-like. A highlight is described as an anomalous intensity value between two equal intensity values. Thus we would have a 'crack' of light between two black areas, or a dark line between two lighter regions.

Although the development is quite complex, it is apparent that this operator can be mechanized without too much difficulty. The intention is the design of a non-local operator, but seemingly valid comparisons are made against local operators by first truncating portions of the new

operator to simulate the local effect. With this adjustment, the operator takes on the following form:

$$EM = R'' + Ka^2 - MIN(R,R') \qquad (17)$$

where:

R is a measure related to the highlight similarity of a region,

R' is a measure related to its edge similarity,

R'' is a measure of its homogeneity,

a is a measure of the local gradient,

K is some constant.

Each term under the MIN function in the above formula represents a different type of edge that the operator is sensitive to. Thus it is seen that this particular operation is positive to, or is concerned with, cliff-type edges, and peaks. Griffith [13,p.25] also mentions a simple generalization scheme to enable profiles of other types of edges to be incorporated into the equation.

Comparisons with the Roberts Operator and the Herskovits Operator using representations of six different edge types show that it is [13,p.25] "...twice as sensitive as the Roberts operator..." and "...slightly more sensitive..." than the Herskovits operator. Griffith mentions that 500 microseconds was used to perform one local operation. Compared to the methods mentioned previously, this is much slower. This statement must, however, be tempered with the knowledge that Griffith's time is

'actual', and is not calculated using the procedure of this thesis. He concludes that his operator has a "...marked superiority...providing the cost of obtaining intensity values is considerably greater than the cost of computation."[13,p.26].

It is of slight concern that he appears to modify the Roberts operator for his comparisons. It is noted that Roberts uses the square root of the intensity values as input to his operator, whereas Griffith uses the natural logarithm of the intensity values as input to the Roberts Operator. Since both operations (ln and sqrt) have similar effects, it is unlikely that the results of that comparison would be made invalid.

II. Quasi-parallel Methods

This section concerns recent methods of edge detection which cannot be classed as local, yet which are exceedingly far from the concept of parallel detection. One can refer to this class of methods as quasi-parallel, meaning that the operator assigns edge indications to more than one pixel with each application.

It was mentioned earlier that parallel processing techniques for edge detection are currently being emphasized over sequential methods. To a large extent this change in

emphasis has come about because of the increasing availability of parallel processors [29], or at least the use of their simulation languages (e.g., PAX II) [32,21]. It is probable that the properties of the mammalian eye have made this continuing change toward parallelism seem logically reasonable, since it is obvious that edge detection, if done in the neuronal network of the retina (as seems to be the case), is done in parallel.

Although one assumes that edge detection is accomplished by some sort of parallel processing within the human eye, being more specific as to detailed techniques is difficult.

The experiments carried out by Lettvin et al [25] on the frog, and by Hubel and Wiesel [19] on the cat have not been duplicated on humans. One can only suppose that the human eye performs edge detection in a fashion similar to that of the lower mammals (a perhaps unreasonable supposition, considering the structural differences). Thus the quasi-parallel methods to follow use techniques not founded only upon the human visual operation (whatever that may be), but instead are based on mathematical heuristics for measuring differences across large areas simultaneously.

Hueckel's Operator

Manfred Hueckel [20] has designed an edge operator which examines an arbitrarily large area of a picture matrix in a single operation. Typically, the number of elements analyzed at one time is chosen from the following: 32,52,69,88,137. These pixels are organized into an approximately circular region. Indeed, each of the numbers just mentioned represents a number of pixels which can be arranged into an array which more closely resembles a circle than some other number within the range of the set. Pingle and Tenenbaum [33] use this method as a coarse basis for their accommodating edge follower.

Using spatial filtering techniques, an edge which appears to pass through the centre of the input 'disc' can be thought of as the zero crossing of a brightness wave. The wave's maximum would appear on one side of the centre, the minimum on the other. Then as far as the operator is concerned, an edge can be detected if the operator is able to produce a matching wave whose orientation, amplitude, and wavelength differs minimally from the input wave(edge). This matching wave must also have absolute characteristics which are above some arbitrary threshold.

It is felt [20] that wavelengths three octaves higher than the diameter of the input disc should be discarded as high frequency 'noise'. With a low pass filter, the eight lowest Fourier components are singled out, and the higher

components are discarded as being representative of noise. In this fashion only edges whose size (width) is comparable to the size of the input disc are allowed to pass. In particular, highlights, cracks, and micro-edges are lost.

Hueckel attempts, through abstract mathematics, to mechanize the following postulates:

a. The importance of the input data decreases towards the periphery of the disc.
b. An operator which locates edges need not be sensitive to noise of high spatial frequency.
c. An operator which locates edges by its nature is sensitive to noise of low spatial frequency.
d. The operator's computing time should be minimal.

A set of four functional equations were developed to satisfy the above postulates. These equations (not yet published) uniquely determine a matrix of constants H(ip). Letting D be the input disc composed of elements p, each of grey level Ep, then:

$$a(i) = H(ip) * Ep, \quad \text{for } i = 0,\ldots,7. \tag{18}$$

One notes that a set of eight numbers is developed. This set is mapped into a 6-tuple (c,s,r,d,b,k), whose elements can be understood intuitively from the following:

a. $cx+sy \leq r$ defines the brighter of two regions whose common boundary is an edge in the input disc.

b. cx+sy>r defines the darker of the two regions mentioned above.

c. The element b is the average grey level on the brighter side of the boundary.

d. The element d is the incremental grey level that must be added to b to equal the average grey level on the darker side.

e. The element k is the 'edge signal-to-noise ratio', i.e., the credence that is assigned to the edge described by (c,s,r,b,d

It is clear that the above 6-tuple defines an edge. The parameters of this edge are produced as a result of minimizing the Hilbert distance between the set of a(i)'s developed from the edge input data and a set of numbers developed from an ideal edge. When the ideal edge and input edge data match closely, the operator outputs (c,s,r,d,b,K).

Hueckel claims a speed advantage over other (unspecified) methods of from 16 to 81 times, and states: "Both its speed and reliability are based on the comparatively large size of the input area."[20,p.5]. As an example of its speediness, it is affirmed [20] that an input disc of 52 pixels can be processed in 1/100 seconds on a PDP-10. This is approximately 10000 CTUs per local operation but, because of Hueckel's method of 'stepping' across the

picture matrix, this time is effectively reduced to 400 CTUs, still considerably greater than previously mentioned methods.

If the time given is correct, then there must exist other reasonable methods that are 16 times (at least) slower. Thus, there must be a method which take at least 6400 CTUs per pixel for edge detection, a suspect result.

Aside from the contradictory speed results, note that the operator is isotropic in operation, and being much more sophisticated than many others, uses more storage for its operation (approximately 100 FORTRAN statements), in addition to the data structure needed to hold edge information (approximately ten percent of the picture storage area).

Schubert [43] has noted that in [20, p. 9], composite edge results (i.e., a line drawing) show that sharp curves, and thin lines are missed by the operator, and abruptness of edges is shown with much less differentiation than the eye can discern. Of course, Hueckel can deal with this problem by using a smaller (and hence higher-resolution) version of his operator in addition to the larger one, but this would aggravate the time problem.

The width of edges detected by the operator, being responsible for its speed (or lack of it), is undoubtedly also responsible for the above deficiencies.

Rosenfeld's Method

Rosenfeld [40] states that "A complete edge detection system should employ, at each point, pairs of neighbourhoods of various sizes and at various orientations; the largest of these should have size comparable to that of the entire picture". This conclusion is based upon the assumption that in most pictures there exist varying 'sizes' of edges, some quite wide, some narrow. To detect a wide edge (e.g., between two large smeared areas), the grey level averages must be determined over relatively large areas in order that a correct decision be made. Thus, large pairs of neighbourhoods are used as data input to a large (or wide) edge detector. An analogous reason could be stated for the justification of small neighbourhoods as input to a small (micro-edge) detector.

A reasonably simple technique is used to implement a parallel processing capability and make use of multi-size neighbourhood operations.

If a picture matrix is shifted to the right one element (i.e., a shift that is analogous to a machine language shift instruction, whereby zeros are brought in from one end of a word, and bits are lost at the other), and then added to itself, the result is that each element is the sum of the original element in its location plus the element in the neighbourhood to its immediate left. If one now shifts this 'summed' picture TWO places to the right, and

again adds the picture to itself, the result is now that each element is the sum of the original element in the location plus the three neighbours to its left. Generalizing, K right shifts and additions suffice to add up the 2**K grey levels to the left of an element (including the element itself). Because the number of terms in each element sum is a power of two, it is very convenient to find its average: a simple left shift of the binary point K places transforms the sum into the grey level average of its constituents.

It should be noted that, as intermediate steps, the grey level sums have also been computed for all neighbourhoods 2**H elements long, $0 \leq H \leq K$ (here H = 0 is the original picture itself). One could, at any stage in the shifting process, compute averages as above. The computation of averages at small K and then later at large K produce the input for the large and small edge detectors respectively.

To compute an array of EMs from the array of averages, a shift of 2**K places, and a subtraction operation is required. The result is an array of differences. This array of differences can then be scanned, suppressing all but the maxima over the matrix; or, the outputs of the large and small edge detectors can be multiplied, which effectively sharpens the edge. Rosenfeld [40] has noted that it is unclear why this multiplying operation should be so effective. In any case,

this penultimate result can then be thresholded, to yield a binary image which indicates edges.

The extension of this method to square neighbourhoods (rather than linear) is made possible by alternating the rightward shifts with downwards shifts, so that the total number of shifts and adds is 2K. The result is that the sum in any element a(i,j) of the array A is the total of all the elements in a 2**K by 2**K square of the original picture P with p(i,j) being the lower righthand corner element of the square. Averaging and thresholding is carried on in a way analogous to the linear case.

This method is characterized by:

a. Efficiency (as a result of parallel processing).
b. A capability to detect a broad range of edge widths.
c. A somewhat lesser capability to detect a textural boundary where the regions concerned may have identical grey level averages.

The benefits of this technique are attained at the cost of:

a. The need for highly sophisticated computer ware, both hard and soft.
b. Some 'squaring off' of curved edges.

c. Increased 'noise' in detected edges, resulting from the use of parallel techniques which, by their nature, cannot use sequential edge-following techniques, with cleaner results. However, this susceptability is probably more than balanced by the operator's capability to 'average out' noise by carrying out computations over large segments of the picture matrix in one step.

It must be finally noted that the efficiency mentioned above for parallel processing can only be fully realized with a truly parallel processor. Simulation of this processor does not provide savings in CPU time.

The use of storage on implementation of Rosenfeld's method would appear to be less efficient than other methods since a requirement exists for three images of the picture to exist at one time, thus limiting the absolute region size for any one examination.

III. Conclusion

Several methods of edge detection have been examined. They range from rudimentary gradient measurements over a tiny picture segment to complex mathematical operations over a large portion of a picture. In several cases it was shown that contradictory results can be obtained, i.e., results that are intuitively incorrect. Whether these wrong indications are statistically serious is a moot point.

Using the least complex versions of the operators, the LaPlacian was the fastest, followed very closely by the Roberts and the Gradient. Farther back came the Herskovits and the Shirai-Tsuji operators. The statistical methods were much slower (e.g., the SML consumed 111 CTUs) since their operations are more complex. If Hueckel's calculations are correct, and understood properly, then his operator is the slowest.

All of the operators have difficulty with a correct isotropic response, but the inclusion of more complex root operations (rather than the faster absolute value operation) would improve this characteristic.

Each of the detectors is partial to specific types of edges. The Roberts favours narrow edges of the step type, the gradient--steep, broad edges, the LaPlacian--thin ridges. The perfect step edge is indicated best by the Herskovits. The SML appears suited to noisy ridge edges.

Analytical judgments on the remainder are difficult.

Storage costs for each of the operators turn out to be relatively unimportant when one considers the size of picture matrix that must be maintained in core memory for edge operations. However, it is noted that Hueckel's method takes 100 FORTRAN statements, which is likely to cramp the capacity of a small computer (which has to accommodate portions of the picture matrix besides).

All of the operators suffer in the presence of noise, but it would appear that Rosenfeld's method can tolerate noise better than the remainder, since it is able to work large segments of the picture at once, and thus 'average out' noise contamination.

Aside from Rosenfeld's method, none of the operators seem to have the flexibility to easily change their segment size dynamically during an edge operation.

The above results lead to what is perhaps an obvious conclusion: determination of the best operator is not analytically possible unless one utilizes a standard picture, and has recognized criteria for determining 'goodness'. Chapter 4 proposes such a picture, and introduces numerical measures to grade the merit of an edge matrix.

It was noted that statistical methods fail to recognize the position of pixels within a picture segment. It seems plausible that the incorporation of some positional measure within a statistical procedure would result in an effective edge detector. Chapter 3 is directed towards the development of such an operator.

Chapter 3

ANOTHER EDGE DETECTOR

The preliminary examination conducted on several edge operators in Chapter 2 indicates that improvements could be made to the standard methods, or possibly that a new and better local edge detector might be useful. This chapter concerns the development of an edge algorithm based on a measure of order found in the regions adjacent to a suspected edge.

I. Foundations

Chapter 1 defined edge detection in terms of the probability that some element p(i,j) was on the border between two regions. Then, given two and only two contiguous regions in one's field of view, it seems mandatory to assume the presence of one edge, and thus at least one edgel. In other words, identifying two and only two regions in one's field of view is equivalent to identifying one edge in that field.

In a one-dimensional binary matrix, such as 0000111100011 0, note by the above statements, that the five binary regions produce four edges. On a more local

examination, to wit: 011; two regions are noted, and thus one edge. Since this is a one dimensional picture only one edgel per edge is possible. We would normally assign the central pixel of the examined area as the edgel.

In the above examples there was no doubt involved in our identification of a region; however, there is also no doubt that the converse is the norm. One is usually uncertain, to some extent, in the categorization of some area as a 'region'. This uncertainty must therefore extend to the identification and labelling of the associated edge. Thus one can say that: the degree of certainty with which one identifies two and only two regions in the field of view corresponds to the degree of certainty that one edge is present.

Further sections in this chapter will develop this degree of certainty into an algorithm for edge detection, both in the binary and the grey level matrix cases. This measure will take the form of a difference sum, DS; whereby the sum of absolute differences of pixel values across a picture segment will be suggested as being inversely related to the probability of the presence of an edge oriented orthogonally to the scan.

II. Region Detection

For the purposes of this chapter those subsets of picture matrices whose grey levels satisfy some spatial

statistical distribution criteria, and whose elements are edge-wise contiguous, will be distinguished as 'regions'.

Binary Regions

A binary region is characterized by the presence of a subset of ones and zeros, whose quantity ratio is prescribed by the statistical constraints for that region. For example, one might insist that the regions contain exclusively ones or zeros, thus forcing a uni-modal distribution over the region. On the other hand, the definition of an even mixture of ones and zeros as characterizing a region would give rise to a balanced bi-modal distribution.

Using a uni-modal distribution to detect edges would require the satisfaction of that criterion over two adjacent areas each with a different binary value. For example, it is an easy matter to:

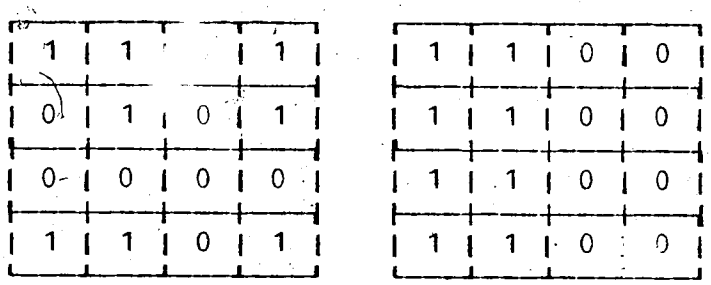
a. Examine one area and determine its distribution as uni-modal in ones, then declare it to be a '1' region.

b. Examine an adjacent area and determine it as uni-modal in zeros, and declare it as a '0' region.

c. Conclude that there is an edge between these two regions.

d. Assign an edge indication to some element(s) on the border between them.

Note that one might observe both of the above areas at once under bi-modal distribution criteria. The confirmation of bi-modality would again seem to indicate the presence of an edge. The obvious danger of this more efficient method is illustrated in Fig. 14 below. Note that both regions are balanced bi-modal, but only Fig. 14b contains one edge. Therefore, confirming a balanced bi-modal



a. Many Edges b. One Edge

Fig. 14. Balanced bi-Modali

distribution is not enough. One must ascertain that each binary level is positionally grouped or clustered (to some arbitrary extent) in order for an edge to be indicated. Chow and Kaneko [3] measure the variance of the neighbourhood surrounding a point to determine bi-modality, but do not check the order of clustering of this neighbourhood.

Dacy and Tung [6,pp.84-85] list three methods for identifying randomness in point patterns. To detect the degree of clustering they prefer their 'Order Method', which amounts to measuring the distance to a point's nearest

k neighbours, for all points in the examined area. The variability of this distance measure is shown to be proportional to the randomness of the distribution.

There are computational and efficiency problems with this method. One must select k properly, define 'distance', and compute this distance kn times. Finally, a variance calculation is done and compared against some criterion of randomness. The CPU time taken for this operation is a function of the density of the neighbours in the pattern, and is thus unpredictable from pattern to pattern.

These problems lead to the proposal of a sub-optimal procedure to indicate the degree of 'randomness' in a point pattern distribution. This disorder is then related to the number of edges in the pattern. The procedure is sub-optimal in that texture shows a high degree of disorder, and will tend to cause edges to be missed.

The Difference Sum Measure of 'Disorder'

The Difference Sum (DS) over an m x m binary sub-matrix B is computed as:

$$DS = HDS + VDS \tag{19}$$

where $HDS = SUM(|b(i,j) - b(i,j-1)|)$, and

$VDS = SUM(|b(i,j) - b(i-1,j)|)$

where, SUM represents discrete summation;

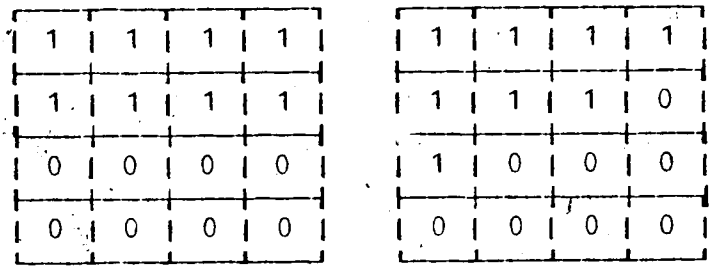
for HDS, i=1,...,m; j=2,...,m; and,

for VDS, i=2,...,n; j=1,...,n.

In words, HDS in (19) is computed by summing the absolute differences between all pairs of horizontally adjacent pixels. In an analogous manner, VDS is calculated.

Considering the extent of clustering or grouping of identical values in a binary matrix as a reflection of the degree of 'order' in the matrix, it can be seen that DS is inversely proportional to that order. For example, consider the one-dimensional matrix 00001111. Here is maximum clustering, reflected by a DS of 1. On the other hand, 01010101 shows minimum cluster size with DS=7. There is an identity relationship, implied by the definition, between DS and the number of micro-edges (not edgels) in the matrix.

Given a balanced bi-modal distribution over an m x m binary matrix, it is clear that DS cannot be less than m. Consider Fig. 15a below. There is no other configuration that will reduce the number of 1-0 pairs to less than m=4. Note that the number of micro-edges formed is identical to DS, and is also a minimum. Fig. 15b shows another balanced bi-modal distribution with DS=6. In this pattern the grouping of 1's is slightly less pronounced. Therefore, 1/DS reflects the compactness of clustering by measuring the length of the border between the two binary values. Given balanced bi-modality, 1/DS will be taken as proportional to the probability of the existence of two distinct adjacent regions within the pattern. From a previous paragraph, two distinct adjacent regions give rise to one edge.



a. Maximum Clustering b. Non-Max Clustering

Fig. 15. Degree of clustering

At this point it is noted that straight diagonally-oriented edges, although intuitively as 'sharp' as vertical or horizontal edges, are not as positively indicated by the DS measure. In fact, the DS computation indicates a diagonal edge to be 1/3 weaker than the same step edge oriented vertically (horizontally). This anisotropy is highly undesirable, and will be largely countered by 'unbalancing' the modality measure when a diagonal edge is suspected (i.e., when VDS=HDS). This unbalancing reflects the difference in the size of two regions separated by a diagonal edge, when the area examined is square.

To measure the balance of a bi-modal distribution one could define some simple linear measure. For example, compute the frequency f of occurrence of one binary value within a pattern of n elements; then the function $(n/2) - |(n/2)-f|$ is linear, and peaks at n/2. However, to

provide a sharper cutoff (and consequently narrower edges), and to accommodate the need for a slight imbalance by diagonal edges, the following non-linear general function is proposed:

$$f(2s - f)$$

where s is the desired region size containing elements of frequency f.

Since the derivative of the above function with respect to f is 2s-2f, it is a maximum at f=s. If one desires an exact balance, then the general formula transforms to f(n-f). Using a 4 x 4 sub-array size, f(16-f) results. For diagonal edges in this same array, a reasonable choice for s is 10. Our slightly unbalanced modal measure then becomes f(20-f).

From the foregoing one can consolidate the formula for the edge merit (EM) of an arbitrary (but usually central) pixel of an m x m sub-matrix B as:

$$EM = f(2s-f)/DS \qquad (20)$$

where s is the desired number of elements within the region, f is the actual frequency derived, and DS is the difference sum measure. Note that the constant 1 in the function 1/DS (mentioned earlier in this section) has been replaced by f(2s-f) to account for DS near zero over a highly unbalanced bi-modal region.

A Computer Algorithm for Binary Matrices

The following procedure will transform the n x n binary matrix B into an edge matrix by determining the existence of a binary edge element (bedgel) centred on an m x m sub-matrix (with origin(i,j)), then indicating the type of bedgel (horizontal(1), vertical(2), or diagonal(3)) at b(i,j) by comparing the vertical and horizontal difference sums (i.e., VDS:HDS). Assuming m=5, an edge at (i+INT(m/2),j+INT(m/2), i.e., (i+2,j+2), would be indicated at (i,j), thus overlaying the binary matrix with the edge matrix, and so conserving storage. The following equation is then derived:

$$b(i,j) = (1or2or3) \text{ if } EM(i+2,j+2) \neq f(2s-f)/DS > T$$
$$= 0 \text{ otherwise;} \qquad (21)$$

for i,j = 1,2,3,...,n-4 (i.e.,n-m+1)

where the ideal threshold $T(hv) < (m^{**}3)/4$ for suspected horizontal (vertical) edges and, $T(d) < m^2(m+1)^2/(8(m-1))$ for suspected diagonals; and where the positive value derived (1or2or3) depends on the orientation of the detected edge (as determined by the ratio VDS:HDS).

With FORTRAN-like notation (e.g., 3*4 means 3 times 4), efficient mechanization of the above procedure results in the following algorithm:

a. Initialize variables:

M=m*m, MMRT=M+(M**.5), T(hv), T(d), EPSLON.

c. Go to first(next) sub-array location, or STOP.

c. Over each m x m sub-array (origin at (i,j))

compute:

f, VDS, HDS, M-f, MMRT-f, DS, T(hv)*DS, T(d)*DS,

and D = m(m+1)/2.

d. b(i,j) = 0

e. If(f*(MMRT-f).LE.T(hv)*DS) Go To b.

f. If(ABS(HDS-VDS).LT.EPSLON) Go To k.

g. If(f*(M-f).LT.T(hv)*DS) Go To b.

h. If(HDS.GT.VDS) Increment b(i,j)

i. Increment b(i,j)

j. Go To b.

k. If(f*(D-f).GE.T(d)*DS) b(i,j)=3

l. Go To b.

Note: The constant EPSLON determines if the prospective edge is likely to be a diagonal. If the m x m sub-array is small (i.e., about 4 x 4), then T(d) is essentially equal to T(hv), and a more efficient algorithm becomes possible.

Examination of the above algorithm shows that b(i,j) will be marked as 0,1,2,3; indicating no edge, horizontal edge, vertical edge, or diagonal edge, respectively. An algorithm similar to the one developed above was implemented in FORTRAN IV and tested on a binary matrix similar in form to the reference picture REFPIC shown in the photographs in the next chapter. The matrix had salt and pepper noise added, and also included various amounts of blurring.

The algorithm was implemented using two different area sizes, and two different thresholds (for noisy and ideal edges). The results of that preliminary test showed that:

a. The edge matrix produced was a reasonable representation of the expected edges.
b. As was expected, the size of object whose edge is detected is a function of the sub-matrix check size.
c. Noise caused minor to moderate errors in the edge matrix.
d. The threshold needs only to be a function of the noise expected.
e. CPU time consumed, for a 3 x 3 sub-matrix size, was 716 microsecs/bixel. In CTUs, the time estimate was 242.

An Application

The binary algorithm developed above may be used in the production of map contour curves. A binary matrix (indicating the z-coordinate levels to be outlined) would serve as input. The output contour (edge matrix) would serve as input to some graphical display, or could be manipulated by pattern recognition routines. These operations would be greatly eased by the intrinsic directionality of the edge markers, since they would serve as valuable input to edge-following algorithms [42].

Other methods of performing binary edge detection, e.g., the Gradient or the Laplacian, are not as easily convertible to different area sizes, or as insensitive to noise (since all the pixels in their sub-arrays are not normally accessed during one local operation).

Grey Level Regions

A grey level region is characterized by integer pixel values whose range over all regions is commonly 0 to 63, and over one region is defined by the prescribed statistical constraints for that region. For example, one might insist that a grey level region be comprised of grey levels whose values are identical, with a resulting sharp uni-modal distribution. On the other hand, one might weaken the criteria for region definition to the extent that the distribution would be mesokurtic-shaped or even flatter.

Arguments similar to those used in the section on binary regions again lead to the proposal of a sub-optimal procedure to indicate the 'degree of randomness' in the statistical distribution of a grey level pattern. This disorder is then associated with the likelihood of the existence of an edge therein. Again, it is textured regions that result in the sub-optimality.

Two versions of the procedure are presented here. The first is measureably faster, and less complicated; however, it lacks in flexibility, and is weak with diagonal edges.

The second, since it is a direct outgrowth of the binary algorithm, must look at each of the grey level values as one of two binary values, with resulting complications in making that decision. However, ease in selecting the balance of the bi-modality desired, and in emphasizing diagonal edges, along with more freedom from noise effects, seem to make this second method worthwhile. For convenience, refer to the first method as the Grey Level Deviation Method, and the second as the Grey Level Modality Method. It is emphasized here that both methods use the Difference Sum postulate explained earlier.

No computer algorithm will be given for the two methods since the mechanization is straightforward and simple.

Grey Level Deviation Method

This algorithm uses two measures:

a. Mean Deviation(MD): is the average difference of a set of n numbers from their arithmetic mean. This measure emphasizes the degree of separation of the elements on one side of the average from those on the other. Given a range of n elements from a through b, the MD maximizes with n/2 a values and n/2 b values. Thus, it peaks on a balanced bi-modal distribution with widely separated modes, and with a given range. This last qualification means that two different ranges could have the

same MD but have a markedly different modality characteristic. Additionally, there is no easy way to 'tune' this measure to peak on diagonal edge distributions. These liabilities, however, are to a degree counteracted by the next measure.

b. The Difference Sum(DS): is, as mentioned previously, the sum of the positive differences across a pattern in the horizontal direction (HDS), and in the vertical direction (VDS). The measure 1/DS = 1/(HDS+VDS) maximizes when the grey level values are identical. Given an exactly balanced bimodal distribution, this measure is large when the elements are, in a positional sense, tightly clustered.

Combining these two measures into equation form we have:

$$\text{Edge Merit(EM)} = \text{MD}/(\text{HDS}+\text{VDS}). \qquad (22)$$

Given an n x n grey level matrix G, the following procedure will transform it into an edge matrix B by determining the existence of an edgel centred on all m x m sub-matrices (with origin(i,j)) over G. The type of edgel will be indicated as vertical (1), or horizontal (2), according to the relative weight of HDS and VDS. Equality between them will require an arbitrary decision.

The edgels will be marked according to:

$$B(i,j) = (1\text{or}2) \text{ if } EM(i+INT(m/2), j+INT(m/2)) = MD/DS > T$$

or equivalently, if Total Deviation $> T*DS*m^2$

$= 0$ otherwise. (23)

This GL deviation method is characterized by:

a. Partiality to step-like edges.

b. The edge-width detected is a function of the size of the sub-matrix used in the scanning process.

c. Weaker responses are derived from diagonal edges than from vertical or horizontal, and thus cannot be absolutely identified.

d. Vertical and horizontal edges are uniquely marked.

e. It is not necessary to dynamically modify the threshold T.

f. Salt and pepper noise tends to drive the EM below threshold thus losing possible edges. This effect could be partially countered by exponentiating the MD prior tc the threshold check.

g. The scan normally used is regular and exhaustive but could readily be made context sensitive (that is, the direction of motion of the scan could be a weighted function of the types of the n previous edgels).

Grey Level Modality Method

This method uses two measures:

a. Modal Measure: f(2s-f), the same measure used for binary matrices, but operable on a transformed subset of a grey level matrix.

b. The Difference Sum: again the same measure used for binary matrices, and also operable on a transformation of the GL matrix.

In essence, this method is identical to the binary algorithm after:

a. The elements of the m x m sub-matrix have been examined (i.e., the arithmetic mean and/or the median and/or the range has been computed).

b. The elements of the sub-matrix have been transformed into one of the binary values according to some criteria determined in sub-paragraph a. immediately above, and in perhaps the n immediately preceding examinations.

Thus, one operates on the resulting simulated binary matrix to produce an edge matrix, as explained in the binary algorithm case.

It is obvious that the crucial phase of this procedure is sub-paragraph b. immediately above. Perhaps the easiest choice is to compute the arithmetic mean of the elements, then assign 1's to those above it, and 0's to

those below. The best transformation is, however, not easily proven. The grey level modality method (using the arithmetic mean as a threshold for the binary decision) is characterized by:

a. The same characteristics as the GL deviation method above, except:
b. Diagonal edges can be detected with about the same strength as vertical and horizontal edges.
c. Salt and pepper noise effects are not as serious.
d. Exhaustive scanning can be transformed into an 8-directional contextual scan. That is, diagonal edges can also be traced. However, this can only be done if a separate structure is available to maintain the grey level matrix in its original form throughout the complete operation.

A Reasonable Test

Analytically, it is difficult to decide which of the above two methods is more worthwhile. One must balance the value of diagonal edge detection, and operation in the presence of noise against the worth of higher speed operation and relative ease of implementation. However, in order to make a reasonable comparison among local methods of edge detection, one of the above methods had to be chosen. Arbitrarily, a variant of the Grey Level Modality Method was implemented as a local, exhaustive edge detector with a 4 x 4 sub-matrix size. It had a secondary threshold input

parameter included to match it with the five other standard methods to be tested in Chapter 4. This method is referred to as the Clustering Operator. The next chapter will see this operator and the five others compared in a functional test on real images input through a TV camera and digitizer to a PDP-9 computer system with disc storage.

Chapter 4

FUNCTIONAL EVALUATION OF EDGE IMAGES

In order to functionally evaluate edge matrices, it is necessary to review standard methods for determining image quality, and ascertain their value in the assessment of edge operator output. This assessment must be based on what future operations are to be performed on this output. This chapter emphasizes that edge matrices are normally operated on by line-fitting algorithms; and thus any measure(s) developed must perform their grading accordingly.

A Digital Picture Processing System (DIPPS) is described, within which tests on six edge detectors are carried out. The test images used are described, as are the programs used to access and display these images and their edge transforms. Visual, numerical, and final subjective conclusions are drawn.

I. Quality of Edge Images

It seems clear that the worth of an edge operator must be based on the binary edge matrices that it produces. That is, one can reasonably deduce that high quality edge matrices emanate from highly effective edge operators. Thus,

the assessment of an operator is based on the average quality of its output.

Discussions on image quality, and objective parameters for its measurement (Rosenfeld [38,p.85] and Roetling et al [37]) emphasize the use of resolution and acutance as measures. Resolution relates to the distinguishability of close objects, while acutance is concerned with the sharpness of edges. As Rosenfeld [38,p.85] points out, these measures are not necessarily interdependent. It seems doubtful that resolution measurements could be used to grade the quality of an edge matrix since, by their nature, all the different 'regions' of an image will be marked (identified) by the grey level 0. Thus, no separation is possible using regions alone. On the other hand, acutance measures the sharpness of edges, and so it seems reasonable to use this parameter as a measure of quality. However, the standard method of implementing acutance is:

$$\int_a^b (df/dx)^2/[\,|f(b) - f(a)|\,]\,dx.$$

Using this equation across a '0001000' edge would result in unreasonable answers for most choices of a and b. Yet a and b cannot be pre-set to prevent this. So, it seems that both resolution and acutance must be reserved for quality measurement on 'real' grey level pictures, rather than edge matrices.

One should assess the quality of an image in relative terms rather than the absolute measures mentioned above. That is, a comparison should be made between the sub-optimal output of an edge operator and an 'ideal' edge matrix. The 'degree of match' between these two could be used as an indicator of the quality of the output edge matrix. In this manner a perfect match would indicate that the output matrix was identical to the ideal matrix, and would consequently get a grading of high (perfect) quality.

The usual method for implementing the 'degree of match' measure is to perform a normalized cross-correlation operation between the two images. However, as Rosenfeld [38,p.77] shows, the normalized cross-correlation (needing many division operations) is not necessary to measure the match between two binary functions. Instead, it is proposed to use the addition of two cross-correlations, the first between the degraded binary matrix and the ideal; the second between the 'negative' of these two functions. If f is the degraded binary image (e.g., an imperfect edge matrix), and g is the ideal representation of this image, then, the sum of two cross-correlation operations:

$$f \otimes g + f' \otimes g'$$

would indicate the degree of match between the two images.

Given two degraded edge matrices E1 and E2, and the corresponding ideal matrix E0, two correlation numbers, C1 and C2, are produced respectively. Given C1 > C2, is it

always true that E1 is a higher quality edge matrix than E2? Elementary tests show that two quite different edge matrices can produce the same correlation numbers. Examination of these matrices with a view to fitting straight line segments makes it clear that the fits will not only be different, but obviously unequal, in quality. Therefore, one must also reject degree of match as the perfect indicator of edge matrix quality.

One is then led to re-define image quality as it applies to edge matrices: the measure of quality of an edge matrix is that characteristic whose measured magnitude is proportional to the computational ease with which the most correct lines can be extracted from it. As Huang et al [18,p.1606] have stated: "Much experimentation needs to be done...[to develop mathematical forms capable of edge error measurement]".

An algorithm has been developed during this research to measure the absolute merit of edge matrices. This procedure has been implemented as the program MERED (note procedural details under The MERED Algorithm, Section IV), and was used to functionally test six edge detectors (see Section 4, Software). In essence, the program accesses arbitrarily-sized sub-arrays over the whole edge matrix, measuring characteristics which are thought to be indicative of 'good' edges. The measurement is done twice, with the second scan being orthogonal to the first. Measurements

include: the number of edge indications in the sub-array, the wiggliness (or noise) of the edge indicators along the line of the scan, the connectivity of the indicators, the thickness of the edge (the number of indications orthogonal to the scan), the number of gaps in the line of edgels, and the relation between the faults for the two orthogonal scans. This first attempt at absolute measurement was quite heuristic, but did show reasonable correspondence with one's intuitive notion of a 'good' edge. The test results reported later in this chapter indicate MERED's effectiveness, and illustrate where improvements could be made.

Since the extraction of lines requires some sort of 'line producer', it is appropriate to examine some line-producing algorithms from which one will be chosen as a reference.

II. Line Producing Algorithms

To produce lines from a set of collinear points, it is helpful to know what types of lines are expected, and with what probability. For example, if one knew that only straight lines are expected from the input, and further, that Probability(length(Lines)>100 pixels) > .7, then the design of a line-producing algorithm could be considerably eased. Straight line input is assumed with only the following justification: much of our artificial world is comprised of straight edges (and they are usually oriented either vertically or horizontally). This straight line input

is normally blurred and noisy to some arbitrary extent. With the categories of input set, it is now possible to examine some algorithms for fitting.

The Hough Transformation

One feasible way to perform the transformation mentioned above is to extract line parameters between all possible pairs of n points in a subset of the edge matrix [7]. The slopes and intercepts developed in this fashion could be averaged to indicate the best fitting line for the subset. However, as Duda and Hart [7] have pointed out, for large n the cost in CPU time could be prohibitive. They propose a variant of the Hough Transformation mentioned by Rosenfeld [38,p.151]. In essence the Hough algorithm uses an array of counters of the same dimension as the edge matrix. For every '1' at the matrix point (x(i),y(j)),a line of counters is incremented in the counter array. This line is defined by:

$$y = y(j)*x + x(i). \qquad (24)$$

It is easily verified that for the collinear points (x(1),y(1)),...,(x(n),y(n)), in the edge matrix, the corresponding lines in the counter array have a common intersection. Thus, this intersection will have a high count. In fact, the count is equal to the number of collinear points. Therefore, a longer line is more easily distinguished than a shorter one.

The difficulty with this method is its anisotropy. The intersection of lines produced from horizontally collinear points recedes to infinity. It is evident that collinearity becomes undetectable in this case. Duda and Hart [7] propose and prove a similar transformation without this defect.

Least Squares Fitting

The use of a least squares fitting procedure to develop line equations for edge matrices seems obvious. Customarily, this problem is treated as the solution of an over-determined system of linear equations of the form:

$$Ax = b \tag{25}$$

with matrix A having more rows than columns. In this particular case, the vector x represents the unknown slope and intercept of the best fitting straight line, with matrix A specifying sample values to be fitted. Using standard calculus techniques a set of normal equations can be developed for solution of the parameters of the fitted line. The solution of these equations is straightforward, and usually iterative.

Because vertical straight lines were postulated as an appreciable portion of the input, the form of the line equations must be adapted to handle this condition. Thus, Roberts [36] solves for the coefficients of

$ax + by = c$ rather than $y = mx + b$.

It is seen that the first form allows vertical lines with

manageable numbers where the second does not. The maintenance of the cumulative sums x, x^2, y, y^2, xy, allows for the continuous solution of the coefficients along with an error measure, when the edge matrix is examined sequentially along a proposed edge.

Other Fits

Aside from the least squares fitting procedure, there are two other less common fitting criteria:

a. Minimize the sum of the absolute differences: MINIABS.

b. Minimize the maximum error: MINIMAX.

Minimizing the sum of absolute differences makes it less likely that noise points will be recognized as data. The MINIMAX algorithm increases this likelihood.

If one is concerned with detecting a 'corner' during the tracing of edgels, then those points past the corner will constitute noise to the current line coefficients. Detecting the presence of this noise is analogous to detecting the corner. Thus, using a MINIMAX fit, and examining the generated error and its sign is a plausible method for locating a corner. It is noted that a least squares fit will not be affected by the corner as soon or as seriously. However, the mechanization of these quick reacting fits is not analytic, easy, or common. Thus, this thesis assumes that least squares procedures will be used for line-fitting.

III. Assessment of Edge Matrices

It was concluded above that the least squares fitting procedure is the most suitable method for selection of straight lines to fit sets of collinear points. It was implied that the eye is not suitable to assess the quality of edge matrices, nor are the standard objective means of measurement. It is affirmed here that edge matrices can only be rated highly if our fitting procedure develops the correct lines from them. Thus, the operators that generate the matrices are graded on the quality of lines they ultimately produce.

A Reference Picture

With several edge operators now in common usage, the prospective user must be given some means to enable selection of the most appropriate. It seems clear that there should be some 'standard' picture on which a particular edge detector can be tested. The results of this test (i.e., the edges or lines generated) should give valuable information concerning the effectiveness of that operator.

This standard or reference picture (REFPIC) must have enough generality to be widely useful, and acceptable. REFPIC must comprise all 4 common edge orientations: vertical, diagonal, horizontal, and curved.

Further, these edges must vary from narrow (sharp) to wide (blurred). Finally, these edges must be shown not only in their ideal form, but also degraded by noise.

A real scene containing all the above edge direction types, in all their forms, would certainly be a rarity. Even if it existed, or could be produced, one still lacks the ideal representation of such a scene, since the real scene would be noisy (if nothing else). Thus, it seems that a test picture must be constructed and ingested digitally to retain its ideal form. The addition of Gaussian noise to a duplicate of such a scene is elementary, and would complete the requirement.

It seems necessary to make the picture flexible under various conditions. Therefore, it is suggested that the number of grey levels used and the extent and type of noise applied be input parameters.

A basic argument against this approach is that, whatever method is used to manufacture a noisy, blurred picture, it is still artificial, and so does not precisely reflect the real world. As a compromise it was decided to design and draw a black-white picture containing various types of edges, and input this picture through the total hardware system (described next section). In this fashion normal degradations through the transmission components would result in an acceptable real picture.

It will be the object of later sections to describe the results of inputting REFPIC to various edge detectors. In addition, a TV image of a human visage, FACE, will be input to the operators to test their reaction to 2-dimensional representations of 3-D bodies.

IV. Test environment

The purpose of this section is to discuss the hardware and software conditions under which the tests on edge detectors were conducted.

Hardware

The major equipments used to input grey level scenes, perform edge transformations, store quantized images, and display pictures includes:

a. TV camera and monitor: A small industrial TV camera (with top-mounted monitor) was used to sense the input scenes. Resolution was specified at 650 lines with S/N ratio at 40 dB.

b. Digitizer: The digitizer is an in-house design with serial A/D conversion to 6 bits. Cycle time is a minimum of 24 microsecs. The digitizer can be windowed to any portion of the complete TV image by keyboard input. During conversion the input scene is sampled once per raster line until a complete image column is assembled in computer core.

c. Computer: A PDP-9 computer was utilized:

(i) to prepare incoming quantized samples for disk storage.

(ii) to transform the quantized pictures stored on disk.

(iii) to prepare quantized pictures and their transforms for display on a storage scope.

The computer has an 8K word memory (18-bit word), with a 1 microsec cycle time.

d. Disk: A 256K word disk was used to store quantized pictures. One 18-bit word can be accessed/stored in 16 microsecs.

e. Storage Scope: The storage scope was used to display pictures and their transforms for analysis and hardcopying (by Polaroid photograph). The 1024 x 1024 display matrix gives a recognizeable and reasonable representation of the corresponding TV image.

f. Cassette Recorder: Tertiary storage of picture processing programs, pictures, and their transforms was a cassette recorder. One track of the cassette (128K words) contained ten separate programs, four 256 x 256 pictures, and several smaller scenes.

Software

Major utility software designed and used for these tests includes:

a. BRING: a program to obtain the output of the digitizer, assemble this output into a coherent structure, and transfer the structure to disk. Individual samples were packed 3 to a word until a complete image column was obtained, at which time a header was attached to the data structure, and transfer to disk began. Columns of the TV image were stored on disk in sequential order.

b. SHOW: a program to display on the storage scope a quantized image stored on disk. This program has the flexibility to:

(i) Display any portion of any picture that can be input.

(ii) Position the picture at any point on the storage scope.

(iii) Magnify any portion of a picture for display (to the extent that one pixel can fill the scope face).

(iv) Change the grey level range prior to display. (i.e., although the hardware supplies 6-bit samples, these can be scaled to lower powers of 2 for display).

(v) Threshold the grey levels in such a fashion that any pixel over the threshold will be treated

as the maximum grey level. This facility enables an edge matrix of 1s and 0s to be easily displayed as full white and black respectively.

Pictures are displayed on the storage scope by illuminating a number of points in direct proportion to the quantized pixel value. For example, if a 256 x 256 image on disc is to be displayed over the complete 1024 x 1024 scope face, then each pixel value determines how many scope points in a 4 x 4 neighbourhood will be illuminated. Which of the points will be turned on was a function of a random number generator. Thus, identical pixels seldom gave the same sub-set of illuminated points. Because of the human eye's logarithmic response to light [14], the number of points illuminated cannot be a linear function of the pixel value. For ease of implementation, a function of the squared pixel value was used to access a table containing the number of points to be turned on.

c. STAT: a program to determine certain statistics of stored images. The minimum, maximum, average, and variance of the quantized image is computed. Additionally, the frequency of each of the pixel values was output so that a histogram construction was possible.

d. MERED: a program to evaluate the merit of edge matrices under certain criteria (note algorithm on following page). The program outputs the number of edgels in the edge matrix, and produces measures proportional to the number of gaps in edges, the amount of blur across edges, and the 'wiggliness' or 'noisiness' of the produced edges. These measures were subject to a double normalization prior to production as a set of tables (see Tables 1 to 10). First, each measured value was divided by the numb of edgels detected by its respective transform, then the L-infinity norm was used on each set of results to produce table values ranging from 0.0000 to 1.0000 (i.e., each set of distinct values was divided by the largest value in the set).

Aside from this utility software, six different edge detectors were implemented as computer programs to be tested. Parenthetically, the utility software detailed above will be used to form the basic set of subroutines of a Picture Operating System(POPS). The hardware and software together represent the Digital Picture Processing System(DIPPS).

The MERED Algorithm

Since MERED (a 'word description' is included in the closing paragraphs of Section II) provided data from which conclusions were drawn concerning the relative merit of the six edge detectors, it seems necessary to include here the essence of the procedure from which the program was built. The following description will discuss operations on only one sub-matrix of the complete edge matrix to be evaluated. Important explanatory notes follow the algorithm (written with a FORTRAN-like notation):

a. Select an arbitrarily-sized m x m sub-matrix M (m should be small compared to suspected edge lengths in the edge matrix). Begin searching for the faults in a prospective vertical edge:

b. Initialize to zero the following variables: BLURS, GAPS, NOISE, CT, SUM.

c. CT = the number of edgels in M(i).

d. If(CT.NE.0) Go To g

e. GAPS = GAPS + 1 note 1

f. Go To m.

g. SUM = SUM + CT note 2

h. BLURS = BLURS + CT - 1 note 3

i. Compute AVPSN(i) (i.e., the average position of the edgels in M(i)). note 4

J. Compare AVPSN(i) with AVPSN(i-1) note 5
note 6

k. From comparison in j. above, compute:

TREND = -1 or 0 or +1 note 7

l. If(ABS(TREND(i)-TREND(i-1)).GT.1)NOISE = NOISE + 2
note 8

m. Increment i.

n. If(i.LE.m) Go To c.

o. FAULTS(vert. edges) = GAPS + BLURS + NOISE

p. Rotate the sub-matrix M 90 degrees and perform steps b. to n. once more, then compute:

FAULTS(horiz. edges) = GAPS + BLURS + NOISE

q. Select the final fault set:

MIN(FAULTS(vert. edges),FAULTS(horiz. edges))
note 9

r. Return to the main calling routine (with SUM, and the final fault set) to access the next (last) sub-matrix M.

NOTES:

1. Of course, if the complete sub-matrix M had no edgels, GAPS would be set to zero by the calling program.
2. SUM is used to normalize fault measurements of edge matrices which may have vastly different lengths of edges. For simplicity, SUM is calculated twice in the algorithm.
3. This reflects the supposition that more than one edgel/line constitutes blurring.

4. For example, consider the row of edgels 1 0 1 1. The average position (AVPSN) would be equal to (1+0+3+4)/3. That is, AVPSN = 2.66....
5. For simplicity, initialization problems are not discussed here (e.g., there is no 0th line).
6. This will yield three logical results: less than, equal to, or greater than (i.e.,-1,0,+1).
7. Obviously, one cannot establish the TREND (orientation) of the prospective edge until at least the second row of M.
8. T' is comparison allows a segment of an edge (in the normally small sub-matrix M) to change direction by 45 degrees without being faulted for noise. Measuring this 'wiggliness' is tantamount to measuring 'salt and pepper' noise. The noise increment '2' was established by experiment.
9. A perfect vertical line of edgels in the sub-matrix gets no faults on a row-by-row scan, but shows up as badly gappy and blurry when scanned orthogonally.

V. Test Parameters

Test Subjects

Two pictures were chosen for testing:

a. REFPIC: a specifically designed black-white picture comprising step edges with various orientations and rates of curvature, and regions of various widths. It is suggested that this design could serve as a model for a much more sophisticated design containing planned blur and noise, along with additional edge types such as roof, ridge, etc. REFPIC was input through the TV camera and digitizer under ordinary office fluorescent lights. A frequency histogram of REFPIC, along with other statistical data, is shown at Fig. 16.

b. FACE: a close-up frontal view of a human face, side-lit with a 40 watt desk-type fluorescent lamp. Focus was exact on the nose. Depth of field was limited due to low light levels and the consequent low F stop. This picture was the only 'real' image tested. Tests on polyhedrals were contemplated but not tried because of time and economic reasons. A frequency histogram of FACE, along with other statistical data, is shown at Fig. 17.

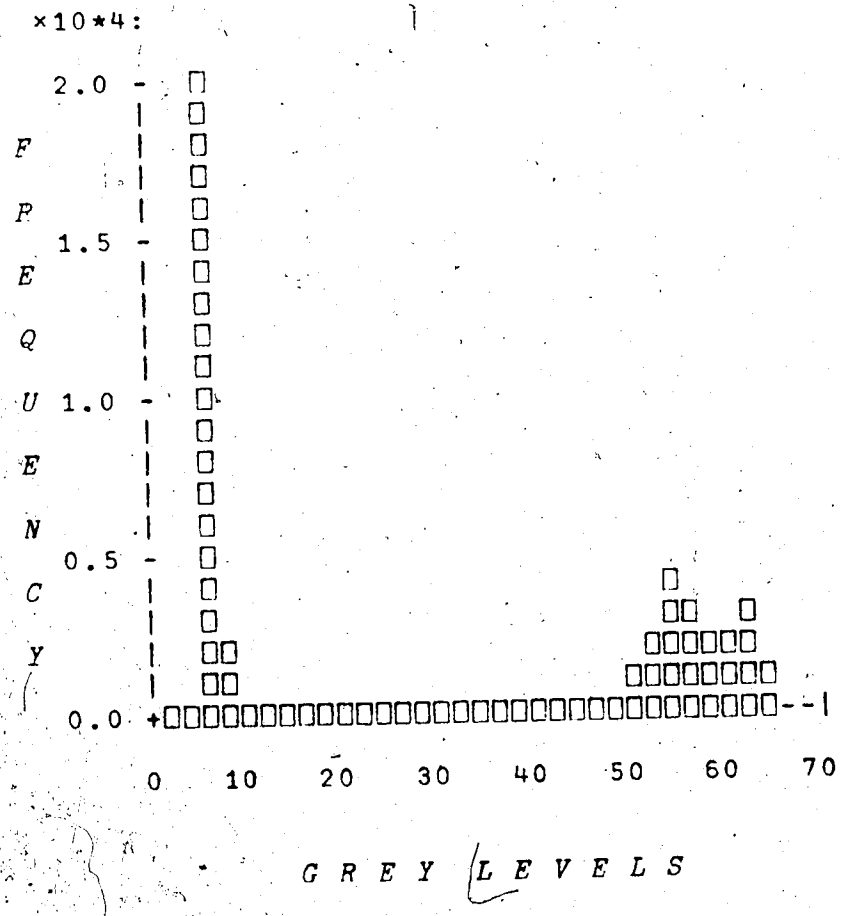


Fig. 16. Frequency histogram of REFPIC
Average grey level: 35.12
Variance: 570
(vertical scale: ×10000)

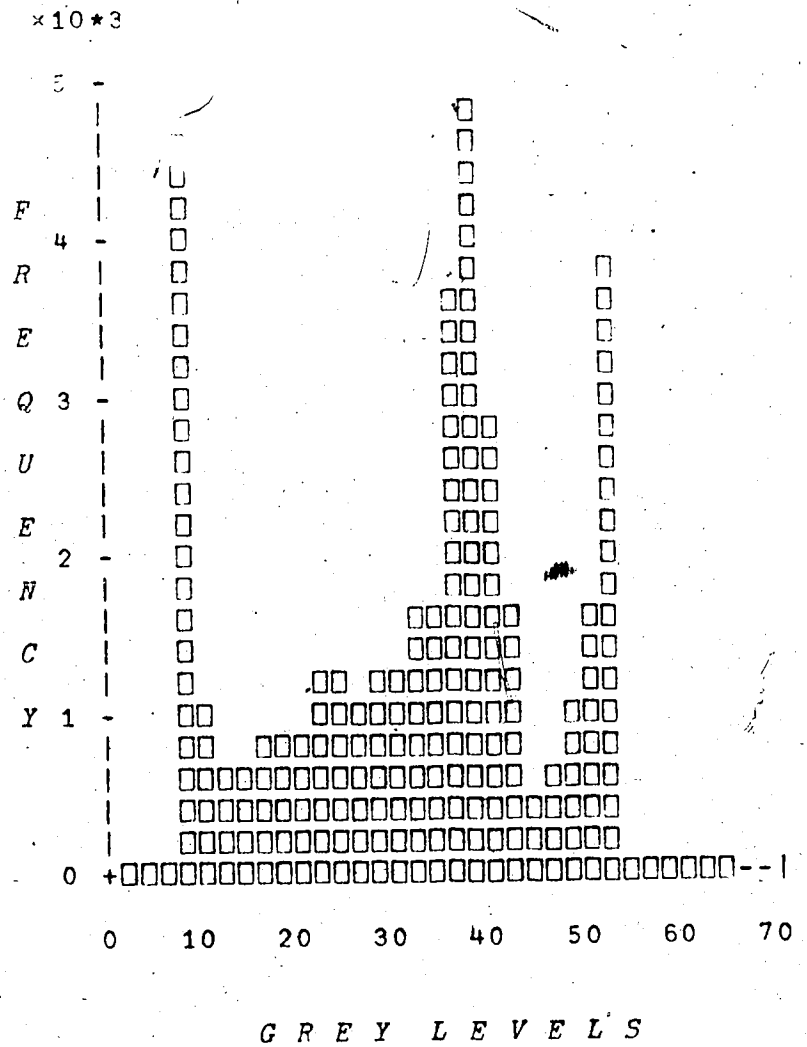


Fig. 17. Frequency histogram of FACE
Average grey level: 30.4
Variance: 210
(vertical scale: ×1000)

Digitizer Conditions

Because the maximum picture size planned for was 256 x 256, the image on the TV monitor covered one-quarter of its screen area. Settings on the digitizer 'aimed' at this area and the picture was sampled. At the time of quantizing the pictures the vertical resolution of the digitizer was twenty percent less than the horizontal, with the result that vertical edges are more blurred than horizontal. This characteristic is somewhat helpful in that the edges of REFPIC have a variety of blur according to their orientation. Minor vertical stretching of the image also results.

Edge Operator Parameters

Rather than 'tune' each detector for its best edge transform (which would require subjective judgment, or an a priori reliance on the evaluation program MERED), it was decided to set the threshold of each so that it would just barely detect a step edge of 8 on a 0 to 63 scale. This magnitude was in accord with the actual steps present in REFPIC. A step edge of 4 was used for thresholding with FACE, which seemed to agree with most of the edge transitions in that picture.

Test Procedure

a. Energize all hardware components.

b. Input and store the two test pictures REFPIC and FACE.

c. Display and photograph the test pictures.

d. Execute the program STAT to record the statistics of each test picture.

e. Run the i-th edge detector program:

(1) Set window and other parameters to conform to input picture criteria of the edge operator.

(2) Produce and store on disc the edge transform of a picture.

(3) Record the time taken for the transformation.

(4) Execute program MERED, and record its evaluation of the edge transform.

(5) Display and photograph the edge transformation.

f. On termination of one test session, dump the contents of the disk onto a cassette cartridge.

g. Continue test sessions at sub-paragraph e. above.

VI. Test Results

Photographic Layout

All photographs in this thesis were taken with a Polaroid Oscilloscope Film Pack camera using Type 107 (speed 3000) pack film. The pictures reflect all phases of

degradation--from the pure image, through all the transmission components to the eye--including: lens aberration, electronic noise in the TV camera, sampling and quantization errors, scope display inadequacies, film graininess, and finally degradation resulting from the photocopying and publishing processes. Visual judgments must be tempered with this knowledge.

So that the reader may easily correlate the photographs with the text, and so that the many references in the text to individual photographs will be short and easily understood, the following notation will be used:

a. REFPIC - refers to the grey level reference photograph at the top of Fig. 18.

b. FACE - refers to the grey level photograph of a face shown at the bottom of Fig. 18.

c. REFPICn - refers to the n-th transform of REFPIC, where n indicates one of the six edge detectors (see Fig. 19).

d. FACEn - refers to the n-th transform of FACE (see Fig. 20).

e. DIAGn - refers to the n-th transform of the diagonal portion of REFPIC in the upper right segment of that photograph (see Fig. 21).

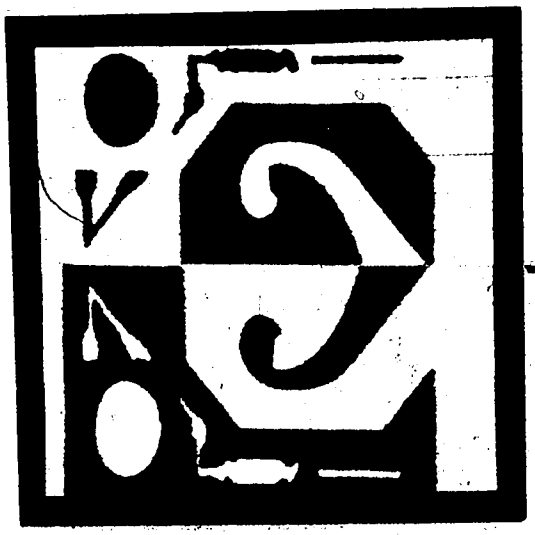
f. CROSSn - refers to the n-th transform of the 'cross' segment in the mid-left portion of REFPIC (see Fig. 22).

g. CORNERn - refers to the n-th transform of the narrow right-angled segment in the upper middle portion of REFPIC (see Fig. 23).

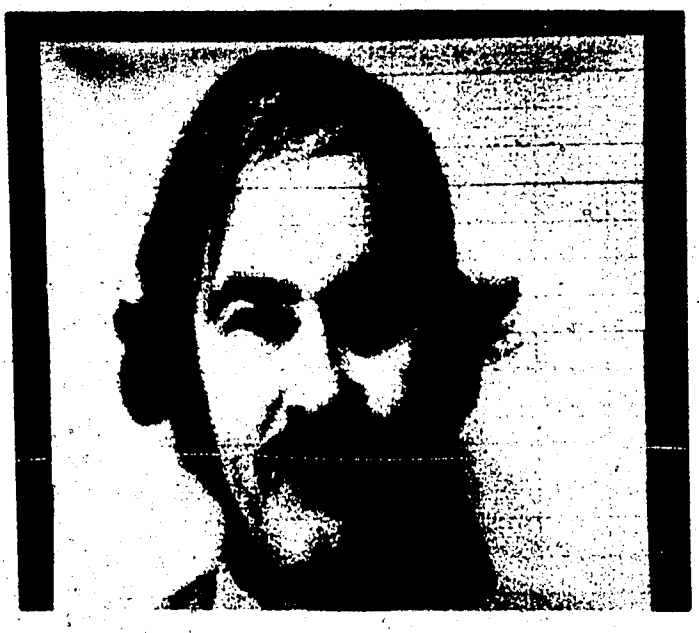
h. CIRCLEn - refers to the n-th transform of the circle in the upper left segment of REFPIC (see Fig. 24).

Fault Tables

The data produced by the evaluation program MERED is shown at Tables 1 through 10. This data, along with edge detector timing data, is used in the numerical evaluation of each operator.

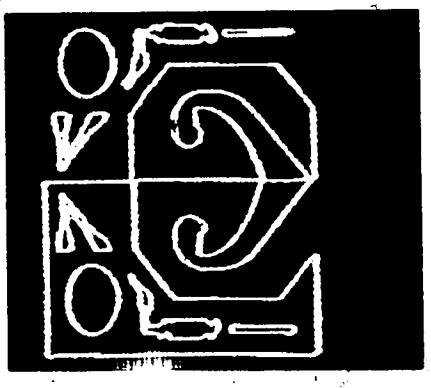


a. REFPIC

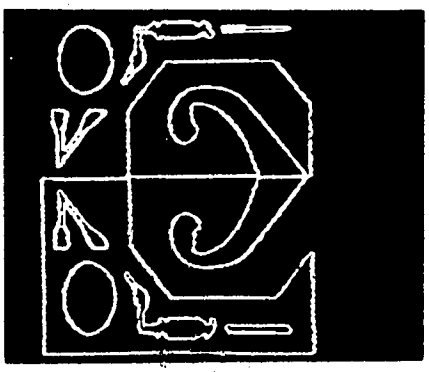


b. FACE

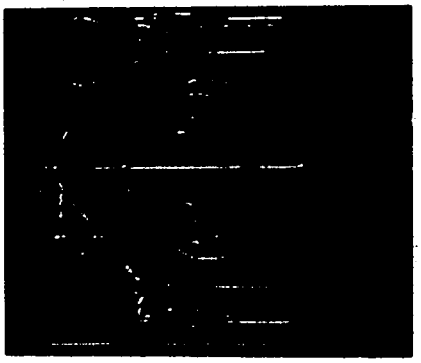
Fig. 18. Storage scope photographs of grey level images.



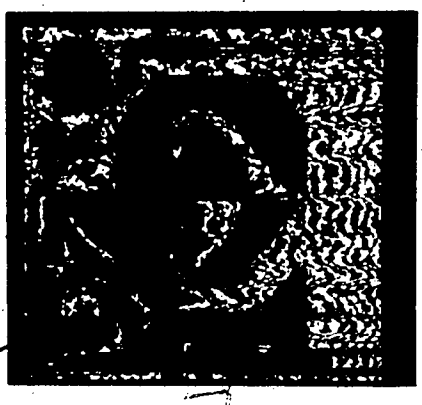
a. REFPIC : oberts Operator



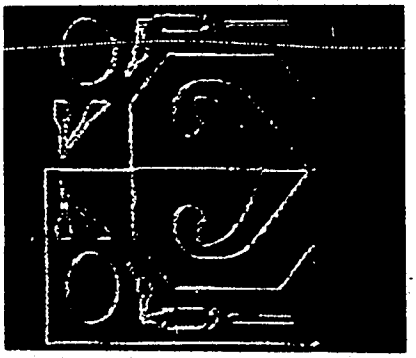
b. REFPIC2: Gradient Operator



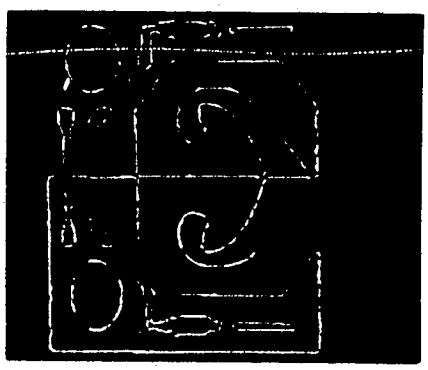
c. REFPIC3: LaPlacian Operator



d. REFPIC4: SML Operator



e. REFPIC5: Herskovits Operator



f. REFPIC6: Clustering Operator

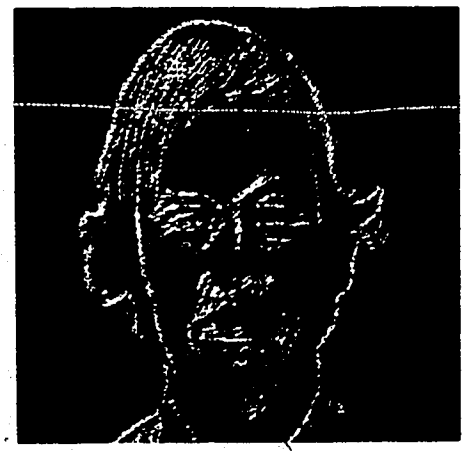
Fig. 19. Edge transforms of REFPIC

a. FACE1: Roberts Operator

b. FACE2: Gradient Operator

c. FACE3: LaPlacian Operator

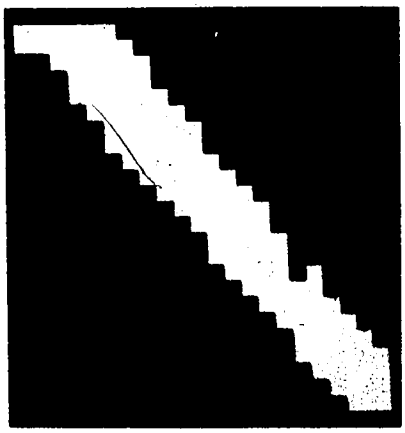
d. FACE4: SML Operator



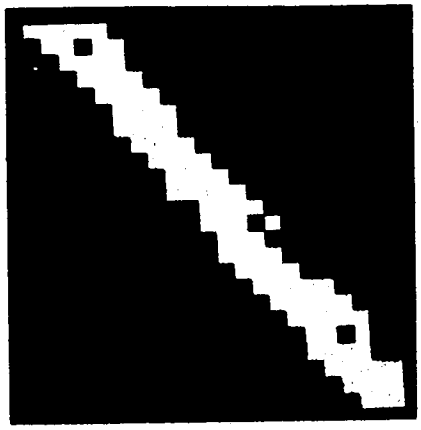
e. FACE5: Herskovits Operator

f. FACE6: Clustering Operator

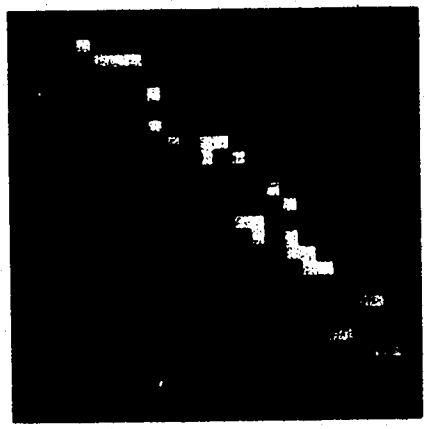
Fig. 20. Edge transforms of FACE



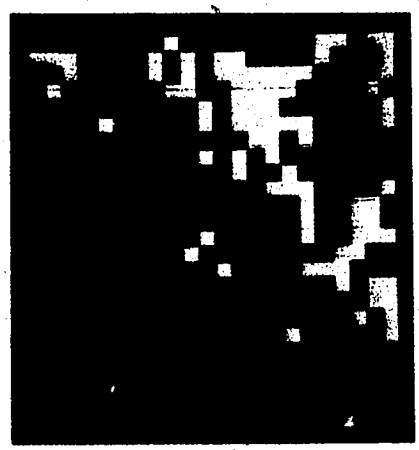
a. DIAG1: Roberts Operator



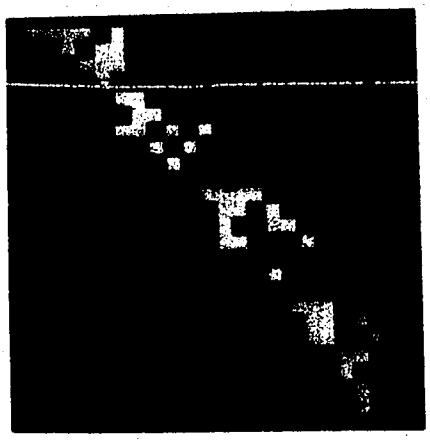
b. DIAG2: Gradient Operator



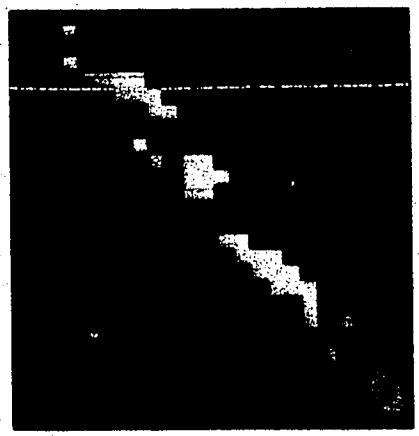
c. DIAG3: LaPlacian Operator



d. DIAG4: SML Operator

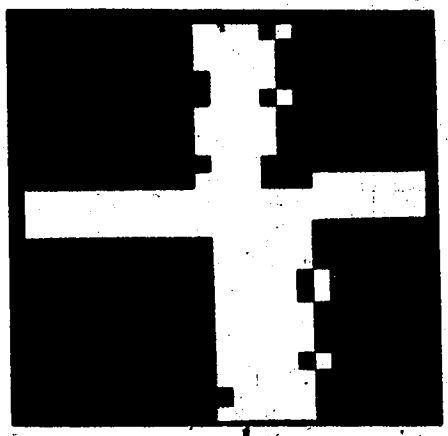


e. DIAG5: Herskovits Operator

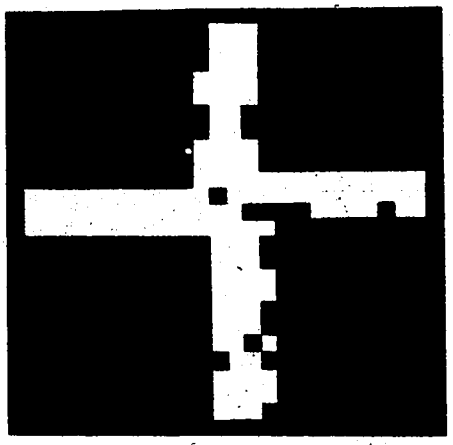


f. DIAG6: Clustering Operator

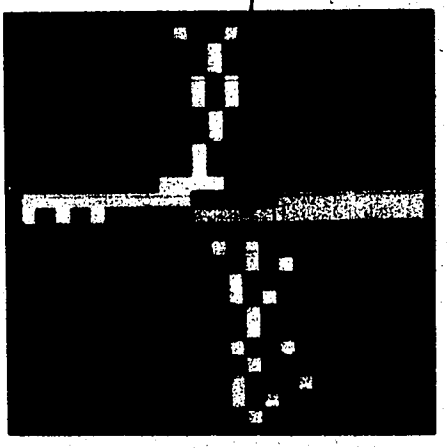
Fig. 21. Edge transforms of DIAG



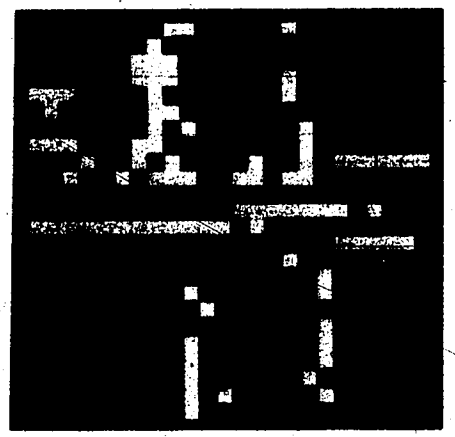
a. CROSS1: Roberts Operator



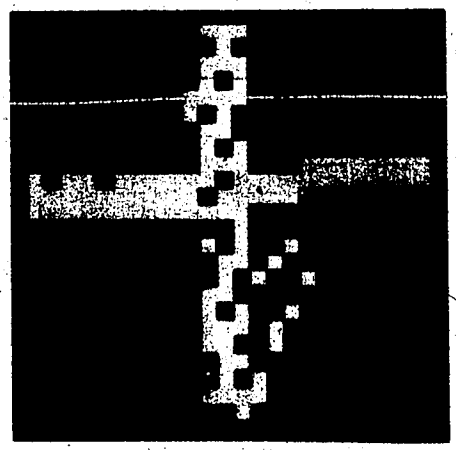
b. CROSS2: Gradient Operator



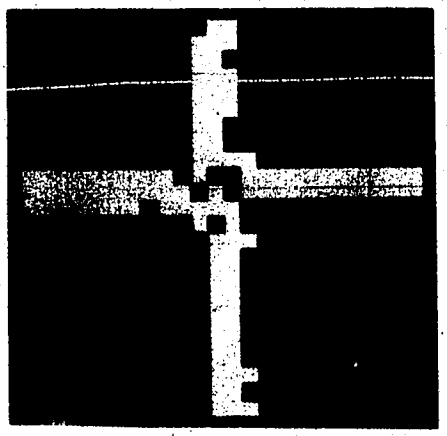
c. CROSS3: LaPlacian Operator



d. CROSS4: SML Operator

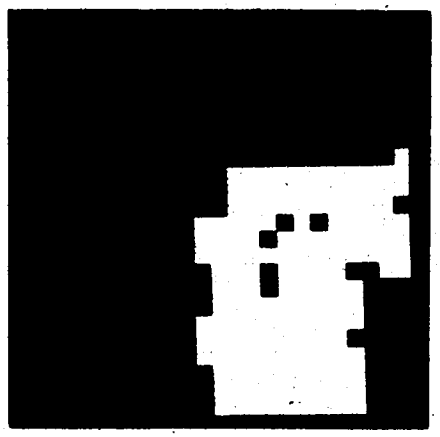


e. CROSS5: Herskovits Operator

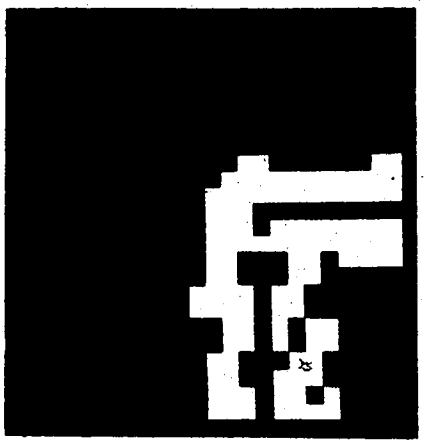


f. CROSS6: Clustering Operator

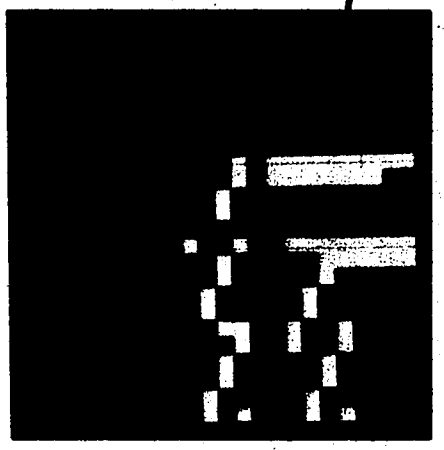
Fig. 22. Edge transforms of CROSS



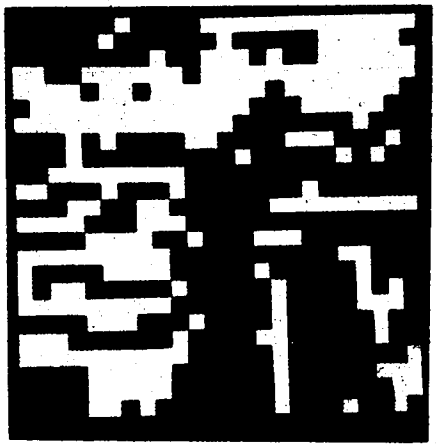
a. CORNER1: Roberts Operator



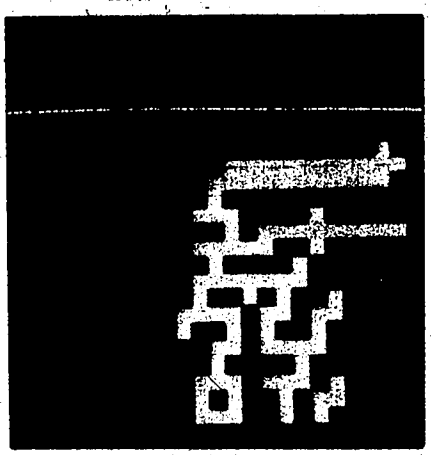
b. CORNER2: Gradient Operator



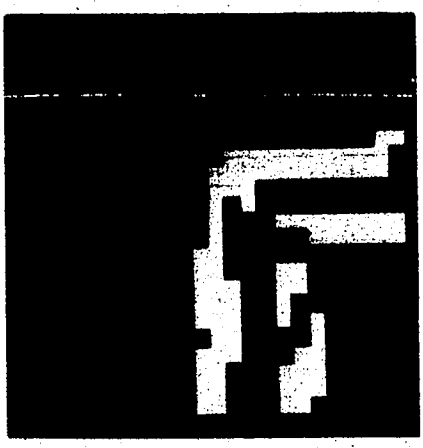
c. CORNER3: LaPlacian Operator



d. CORNER4: SML Operator

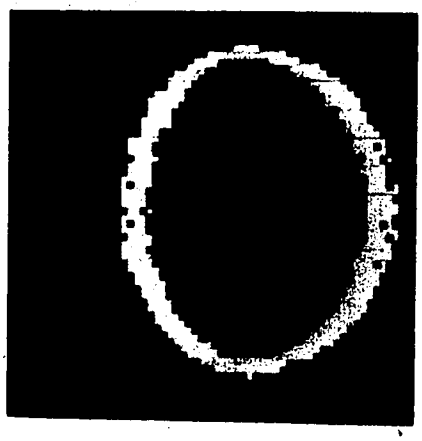


e. CORNER5: Herskovits Operator

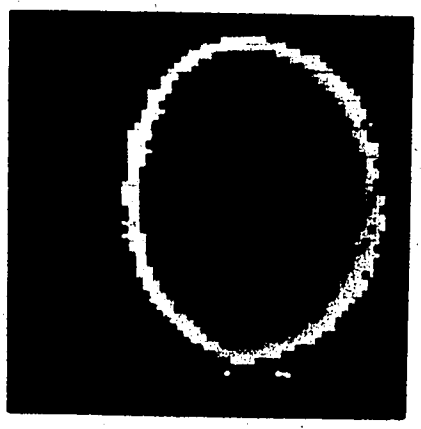


f. CORNER6: Clustering Operator

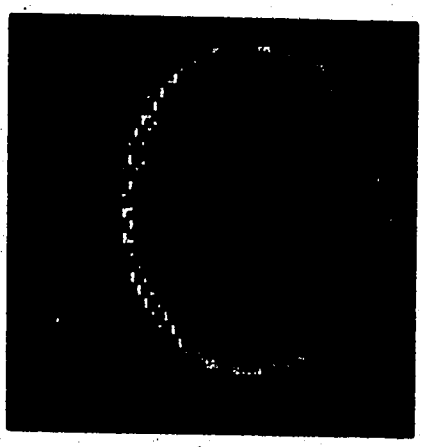
Fig. 23. Edge transforms of CORNER



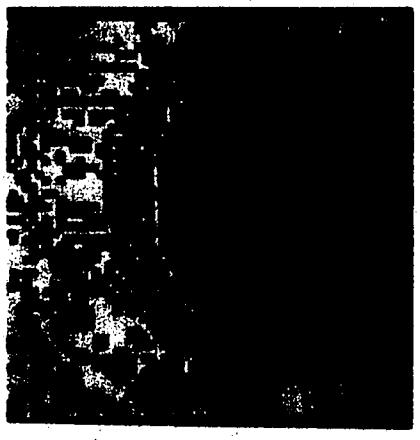
a. CIRCLE1: Roberts Operator



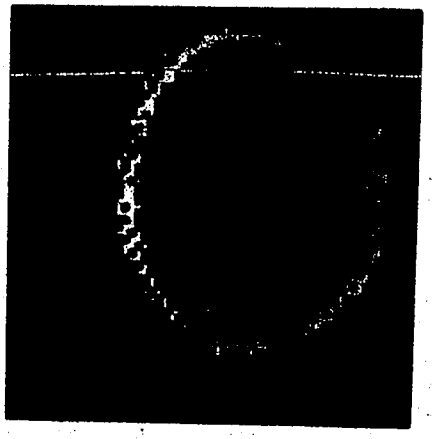
b. CIRCLE2: Gradient Operator



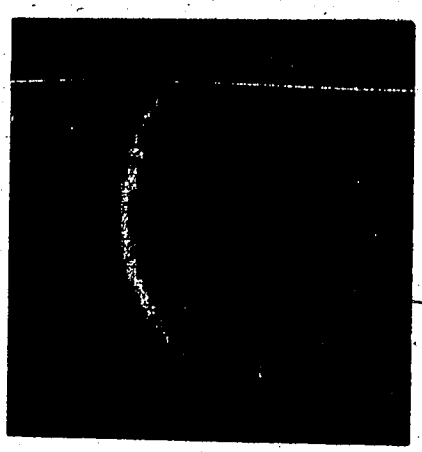
c. CIRCLE3: LaPlacian Operator



d. CIRCLE4: SML Operator



e. CIRCLE5: Herskovits Operator



f. CIRCLE6: Clustering Operator

Fig. 24. Edge transforms of CIRCLE

TABLE NO. 1 .. FAULT TABLE- COMPLETE IMAGE OF REFPIC
(NEIGHBOURHOOD CHECK SIZE = 4)

EDGE OPERATOR	NUMBER OF EDGELS	FAULT TABULATION		
		BLURS	GAPS	NOISE
ROBERTS	10749	1.0000	0.1863	0.0881
GRADIENT	8533	0.8993	0.2777	0.2278
LAPLACIAN	3874	0.4101	1.0000	0.9651
STATLAP	16450	0.7339	0.5650	1.0000
HERSKOVITS	5983	0.6564	0.5087	0.8998
CLUSTERING	5621	0.6950	0.4386	0.4168

TABLE NO. 2. FAULT TABLE- COMPLETE IMAGE OF FACE
(NEIGHBOURHOOD CHECK SIZE = 4)

EDGE OPERATOR	NUMBER OF EDGELS	FAULT TABULATION		
		BLURS	GAPS	NOISE
ROBERTS	11973	1.0000	0.1351	0.2945
GRADIENT	5005	0.8100	0.2627	0.4450
LAPLACIAN	2731	0.3484	1.0000	1.0000
STATLAP	23031	0.8177	0.1761	0.8473
HERSKOVITS	6949	0.6208	0.4740	0.9425
CLUSTERING	3889	0.7450	0.3179	0.2318

TABLE NO. 3. FAULT TABLE- DIAGONAL(TOP-RIGHT OF REFPIC)
(NEIGHBOURHOOD CHECK SIZE = 4)

EDGE OPERATOR	NUMBER OF EDGELS	FAULT TABULATION		
		BLURS	GAPS	NOISE
ROBERTS	130	1.0000	0.1508	0.0000
GRADIENT	99	0.9356	0.1980	0.0000
LAPLACIAN	28	0.2321	1.0000	0.0000
STATLAP	85	0.6309	0.4612	0.2078
HERSKOVITS	53	0.5519	0.3434	1.0000
CLUSTERING	43	0.5291	0.6186	0.4109

TABLE NO. 4. FAULT TABLE- CROSS(MID-LEFT OF REFPIC)
(NEIGHBOURHOOD CHECK SIZE = 4)

EDGE OPERATOR	NUMBER OF EDGELS	FAULT TABULATION		
		BLURS	GAPS	NOISE
ROBERTS	182	1.0000	0.0190	0.0000
GRADIENT	136	0.8922	0.0509	0.0000
LAPLACIAN	69	0.5595	0.6020	0.6522
STATLAP	90	0.4085	1.0000	1.0000
HERSKOVITS	110	0.8523	0.2832	0.4091
CLUSTERING	106	0.7804	0.0980	0.0000

TABLE NO. 5. FAULT TABLE- CORNER(TOP-MIDDLE OF REFPIC)
(NEIGHBOURHOOD CHECK SIZE = 4)

EDGE OPERATOR	NUMBER OF EDGELS	FAULT TABULATION		
		BLURS	GAPS	NOISE
ROBERTS	151	1.0000	0.2649	0.0000
GRADIENT	109	0.9375	0.0917	0.0000
LAPLACIAN	60	0.5084	1.0000	0.5222
STATLAP	244	0.7751	0.4098	0.5137
HERSKOVITS	94	0.7302	0.5851	1.0000
CLUSTERING	86	0.6385	0.3488	0.0000

TABLE NO. 6. FAULT TABLE- DIAGONAL(TOP-RIGHT OF REFPIC)
(NEIGHBOURHOOD CHECK SIZE = 8)

EDGE OPERATOR	NUMBER OF EDGELS	FAULT TABULATION BLURS	GAPS	NOISE
ROBERTS	130	1.0000	0.1185	0.0000
GRADIENT	99	0.9460	0.2263	0.0000
LAPLACIAN	28	0.3994	1.0000	0.3795
STATLAP	85	0.8058	0.1976	1.0000
HERSKOVITS	53	0.7121	0.3698	0.4009
CLUSTERING	43	0.6177	0.5209	0.7413

TABLE NO. 7. FAULT TABLE- CROSS(MID-LEFT OF REFPIC)
(NEIGHBOURHOOD CHECK SIZE = 8)

EDGE OPERATOR	NUMBER OF EDGELS	FAULT TABULATION BLURS	GAPS	NOISE
ROBERTS	182	1.0000	0.0000	0.0495
GRADIENT	136	0.9587	0.0000	0.1324
LAPLACIAN	69	0.5512	0.7609	0.2609
STATLAP	90	0.5735	1.0000	1.0000
HERSKOVITS	110	0.8643	0.0000	0.2455
CLUSTERING	106	0.8457	0.0000	0.3396

TABLE NO. 8. FAULT TABLE- CORNER(TOP-MIDDLE OF REFPIC)
(NEIGHBOURHOOD CHECK SIZE = 8)

EDGE OPERATOR	NUMBER OF EDGELS	FAULT TABULATION BLURS	GAPS	NOISE
ROBERTS	151	1.0000	0.0889	0.2980
GRADIENT	109	0.8889	0.0000	0.0000
LAPLACIAN	60	0.6082	0.2238	1.0000
STATLAP	244	0.8870	0.0000	0.9836
HERSKOVITS	94	0.8166	1.0000	0.6383
CLUSTERING	86	0.7609	0.9369	0.1744

TABLE NO. 9. FAULT TABLE- CIRCLE(TOP-LEFT OF REFPIC)
(NEIGHBOURHOOD CHECK SIZE = 4)

EDGE OPERATOR	NUMBER OF EDGELS	FAULT TABULATION		
		BLURS	GAPS	NOISE
ROBERTS	520	1.0000	0.2153	0.0951
GRADIENT	388	0.9271	0.2492	0.0000
LAPLACIAN	173	0.2985	1.0000	1.0000
STATLAP	957	0.8150	0.3642	0.4648
HERSKOVITS	278	0.7046	0.4759	0.5334
CLUSTERING	262	0.7613	0.2913	0.0943

TABLE NO. 10. FAULT TABLE- CIRCLE(TOP-LEFT OF REFPIC)
(NEIGHBOURHOOD CHECK SIZE = 8)

EDGE OPERATOR	NUMBER OF EDGELS	FAULT TABULATION		
		BLURS	GAPS	NOISE
ROBERTS	484	1.0000	0.1912	0.0758
GRADIENT	383	0.9096	0.2207	0.0479
LAPLACIAN	165	0.3980	1.0000	1.0000
STATLAP	766	0.9487	0.2890	0.5385
HERSKOVITS	272	0.7401	0.5474	0.5055
CLUSTERING	262	0.7193	0.4608	0.1050

Interpretation Of Results

The results of the testing carried on for this thesis can be classified under three headings: subjective visual, numerical, and final subjective judgements (based on the first two of the three). The relative value of each is certainly arguable, viz: the human eye can examine a photograph of an edge transform and opine: good, bad, indifferent, etc. However, edge matrices are not meant to be looked at, but instead, are to be mathematically operated on by line-producing algorithms. In any case this thought does not negate the value of the eye.

The output numbers from MERED (see Chapter 4, Section IV, Software) give a numerical representation of the value of each edge matrix. Though the MERED algorithm used is not analytical, it does not violate standards set for measuring absolute merit of edge matrices (since none seem to exist). But it can be argued that the algorithm is possibly faulty since it is grossly heuristic, and gives wrong answers to some special cases. Countering this, MERED has been checked on ideal and degraded versions of the same edge matrix with satisfactory results. The operating time of each detector was determined using the real-time clock of the PDP-9 computer. The times recorded versus the analytical times developed in Chapter 2 for the various operators show a wide disparity. This is due to the array indexing and 'housekeeping' operations within each detector; however, the

times in Chapter 2 are mostly correct if considered as relative values among the operators.

Final subjective judgements on overall merit are necessitated by the inadequacies of visual and numerical means alone. One must balance the photographs against the numbers, and produce some assessment. Therefore, most of the remainder of this chapter will comprise opinions on the relative merit of each detector, based on the 'hard' results in the accompanying figures and tables, tempered by subjective judgment.

Each of the operators will be assessed separately using relevant criteria listed under Section II of Chapter 1. A judgment of the relative worth of each operator will be made in the closing paragraphs of this chapter.

The Roberts Operator

The version used for this test was a slightly modified version of the one mentioned in Chapter 2:

$$EM = |a-c| + |c-b|. \qquad (26)$$

The Roberts Operator consumed approximately 80 microsecs/pixel in edge detecting, and thus was by far, the fastest of the operators tested. Its speed resulted from simple operations (subtraction) on only 4 pixels, and more importantly, the ease of 'housekeeping' on adjacent elements in neighbouring columns of the picture matrix.

Visual

Examination of CROSS1 shows reasonable isotropy (anticipate slightly wider vertical edges than horizontal because of the disparity between the vertical and horizontal resolution). A comparison among all the FACE transforms shows that the Roberts is quite noisy. This same comparison illustrates its wide dynamic range (i.e., the forehead shadow line is detected where the edge transition step is weak, and yet edges with fairly large transitions are also detected). One also notes its capability to detect narrow highlights (or equivalently, cracks of darkness) by seeing the detected hair fuzz under the left ear of FACE1.

Numerical

The fault tables show definitely that the Roberts is severely degraded by blurring, and as might be expected, has the fewest gap faults. Erratic linearity of edges (called NOISE in the tables) is certainly not a characteristic of the Roberts. In fact, it is the most highly rated in this respect.

The Gradient

The Gradient was implemented in its fastest form:

$$EM = |d-f| + |b-h|. \qquad (27)$$

The Gradient Operator took 167 microsecs/pixel for detection, and thus was the second fastest of the operators. The fact that this detector operated at only half the speed

of the Roberts is due solely to array indexing inefficiencies on three columns of a picture matrix, where the pixels considered were not adjacent.

Visual

All the partial transforms of this operator (DIAG2 through CIRCLE2) show that it approaches isotropy quite closely. The REFPIC transforms (REFPIC1 through REFPIC6) show clearly that the Gradient has less than average noise. The weak forehead shadow edge does not show up in FACE2, thus illustrating that microedges are not easily detected by the operator. DIAG2 shows a reasonably thin diagonal edge with excellent continuity (no gaps). CROSS2 through CIRCLE2 confirm this. REFPIC2 seems the most pleasing to the eye of all the REFPIC transforms.

Numerical

In general, the fault tables show that the gradient blurs edges, but has very few gap faults. In addition, it rates fairly high for linearity of edges, since its NOISE faults are among the lowest. Using a sub-matrix check size of 8 does not change this conclusion.

The LaPlacian

The version of the LaPlacian used for the tests was:

$$EM = 8e - (a+b+c+d+f+g+h+i). \qquad (28)$$

Although at first one might expect the LaPlacian to be equally as fast as the Gradient, the LaPlacian, at 255 microsecs, was quite a bit slower. This is, of course, due to it accessing twice as many pixels as the Gradient, but using the same neighbourhood size. However, it is appreciably faster than the average, since the operations it performs on its neighbourhood pixels are 'one-time' and simple.

Visual

Transforms DIAG3 through CIRCLE3 show that this operator is fairly isotropic. The distinguishing characteristic among all the LaPlacian transforms is their visual faintness. This is due to the operator's capability to 'sharpen' the edges it detects such that there are very few edgels marking detected edges. Of course, the lack of narrow ridge edges in the pictures (except the forehead hair strands in FACE3) probably accentuates this faintness. Additionally, the LaPlacian is reasonably immune to noise (i.e., microedges) as shown by a comparison among all the FACE transforms. The difference between vertical and horizontal resolution, and the LaPlacian's reaction to that characteristic, is shown dramatically in CROSS3. This emphasizes the intolerance of the operator to blurred edges. Its ability to detect around 'thin' corners is quite good, as CORNER3 illustrates.

Numerical

Generally, one notes that this operator has a very good blur fault measurement, and conversely, rates very badly with reference to gappiness. The conclusions to be gained about NOISE are somewhat contradictory in that Tables 3 and 10 rate the operator highly, whereas the remainder of the tables generally rate it low, and sometimes very low. Subjectively, one tends to give more weight to the majority.

The Statistical Modified LaPlacian (SML)

Implementation of this detector was exactly as explained in Chapter 2:

$$EM = (f-Av)/s. \tag{29}$$

It consumed 617 microsecs per pixel, and thus was among the slowest of the operators tested. This time can be attributed to the performance of fairly complex operations on a large number of pixels. Additionally, each of the pixels had to be accessed twice.

Visual

Although this algorithm produced the worst visual results, there were some interesting features. Aside from the overwhelming noise, the edges of FACE4 are the sharpest of all the FACE transforms. In particular, one notes that the mouthline is one edgel wide, and is complete across the full extent of the mouth. Similarly, faint edges in the internal portions of the right ear (not even visible in the

grey level FACE) are outlined very sharply. An unusual feature shows up at REFPIC4, and is accentuated in the partial transform CIRCLE4: note that the edge obtained is very sharp (though certainly gappy), but more importantly, the line of edgels is separated from the noisy regions by a distinct (and relatively wide) black background. It thus seems possible to separate the true edges from the noise with reasonable ease. How this could be done algorithmically is an interesting and probably solvable problem. Turning now to the REFPIC transforms, one is startled by the perfect vertical edge in the lower left of REFPIC4, equally as sharp as the horizontal edges in the middle and bottom of the same photograph. Thus it is seen that, for some light intensity transitions across edges, this operator is hardly affected by the differences in edge blurring. It is probable that the average light intensity across the edge has much to do with this characteristic. In accord with this proposition, one notes the very structured 'noise' in the right portions of REFPIC4. It is likely that this parallel wavy set of edgels is in fact marking true edges, which are not visible to the eye due to its logarithmic response. It is to be noted that even the original image did not show these edges. So it seems that the SML can detect micro-edges when the average light intensity is high, but has difficulty with low averages.

Numerical

The fault tables indicate a medium amount of blurring, a lesser amount of gappiness, and a medium to large amount of noise. This matches what the eye tells us from the transforms. There is no significant difference between the results from the two different check sizes used.

The Herskovits Operator

This edge detector was implemented as explained in Chapter 2:

$$EM = MAX(|2(b-c)| - |a-b| - |c-d|, \quad |2(f-c)| - |e-f| - |c-g|). \tag{30}$$

This operator took 401 microsecs/pixel for edge detection, and consequently was almost exactly on the average of all tested operators. Even though it only accessed 7 pixels in its neighbourhood, it performed a slightly more complex operation on them with the resulting extra time comsumption.

Visual

The line of edgels marking vertical and horizontal edges (see CROSS5) are greatly different in quality, thus reflecting the operator's reaction to differences in the sharpness of edges. Its susceptability to noise is evident. Sharpness on horizontal edges is illustrated by the mouthline of FACE5. The operator is moderately blurry and

gappy on step edges (see DIAG5) between fairly large regions, but is completely confused on crossing a narrow region (see CORNER5).

Numerical

The fault tables show generally that the Herskovits operator has a slightly higher than average susceptibility to blurring and noise, but less than average gappiness. No significant difference was found between the two check sizes.

The Clustering Operator

Implementation of this detector was guided by the characteristics of the grey level modality method (see chapter 3), where the arithmetic mean of the neighbourhood picture elements was used to determine the balance of the bi-modality:

$$EM = f(2s-f)/DS. \qquad (31)$$

This operator was by far the slowest. It took 874 microsecs/pixel to detect edges, which was ten times slower than the Roberts. This big differential in time consumption is due to several factors, the most important of which are: it accesses four times the number of pixels per local operation as the Roberts (and about twice as many as the rest), it must access these elements twice during each operation, and finally, it was designed with more generality

in mind than simply local edge detection (e.g., the neighbourhood size can be an input parameter to the operator). Additionally, 'housekeeping' chores proved costly in CTUs.

Visual

The clustering operator illustrates excellent isotropy by marking vertical and horizontal edges equally well (see CROSS6), even faced with weak vertical resolution. Analogously, one notes that it is not so susceptible to edge width differences. Its noise discrimination is good (see REFPIC6), and its sharpness is very good (see CROSS6). The angle corner of CORNER6 is marked excellently. The operator has moderate visual gappiness.

Numerical

The numerical results for this operator were the most equivocal, but in general they imply that the operator has moderate blur faults, average gap faults (except over narrow regions, where it excels), and is moderate to good with reference to noise faults.

VII. Conclusions

Regarding blurring, both visually and numerically the LaPlacian rates the highest, and the Roberts is the lowest.

The Gradient and the Roberts operators offer the fewest gap faults from their grey level input.

Numerically, non-linearity of edges, i.e., noise, is a characteristic of the statistical modified LaPlacian, with the ordinary LaPlacian and the Herskovits being affected in slightly lesser amounts. However, visual examination of the edge matrices does not present these differences this sharply. This measure of 'wiggliness' or 'noisiness' must be further examined for its fidelity.

The speed of an operator is one of its most important characteristics. Note that the Roberts is the fastest, and the Clustering Operator is the slowest. It should be stated though, that this operator has a built-in potential to move across (and down) a picture matrix in steps equal to one-half the sub-array size used. Since a 4 x 4 size was implemented, such a stepping procedure would have resulted in an approximate four-fold reduction in CPU time consumed. The resulting time useage would be quite competitive with the remaining operators.

In conclusion, it becomes obvious that more complex operations involving more neighbourhood pixels gives generally better results. However, one must balance one's wish for the 'best' operator against the cost in time and complexity necessary to approach this ideal.

Chapter 5

HARDWARE EDGE DETECTION

During the past two years electronic devices have been developed which make it economically feasible to propose realistic designs for hardware edge detection systems. This chapter will give a brief explanation of these devices, and suggest a method of implementation.

The Major Problem

In all of the sequential edge detection schemes presented in Chapter 2, the requirement was to perform some mathematical operation on a small neighbourhood of pixels and produce an indication of the likelihood of an edge being present. If one contemplates performing this operation using hardware working directly on the TV signal (as it becomes available), then a difficult problem arises: how does one access m neighbouring picture resolution elements on each of n different neighbouring lines of a TV raster frame, and all at the same time? It seems clear that the only answer to this severe timing problem is to first delay the k-n+1, k-n+2,..., k-1 th lines ($k \geq n$) until the k-th line becomes available. If this can be done, then TV signals from n different lines (representing the lines' grey levels) can

be made simultaneously available. With the addition of m-1 shorter delays on each of these lines, m TV signals/line become available as input to prospective hardware edge detectors. For simplicity, the interlacing characteristic of a TV raster scan is ignored during the following exposition. Relevant hardware delay methods are now examined.

I. Delay Devices

Analog Delay Lines

Two basic delay lines seem worthwhile mentioning in the context of analog image processing applications: the crystal delay line and the charge-coupled device (CCD). This last device is a product of recent MOS technology.

Crystal Delay Lines

Crystal delay lines operate on the piezo-electric principle in that electrical signals are transduced into mechanical vibrations which are transmitted through the crystal lattice. During their transmission through the crystal, the vibrations travel at speeds much slower than the electrical vibrations. This speed reduction means that the re-transforming of the mechanical vibrations into electrical waves results in an effective delay of the signal. Recent developments in delay lines have shown [22] that electrical signals can be transduced into surface Rayleigh mechanical vibrations in the crystal such that

the delay can be greatly extended without loss of much fidelity in the re-transformed signal.

Corning Glass Works has recently released specifications and costs of one such device [4,5], using ultrasonic glass as the crystal medium. It is the MCA-5.04 Video Delay Line. It can be obtained with delays ranging from 30 microsecs to 64 microsecs, with tolerance of ±3.5 nanosecs. The lowest cost of such a device is currently 250 dollars each. The signal-to-noise ratio is 50 dB, and 4.5 MHz is the 3 dB bandwidth.

The difficulty with these delay lines is that the delay is fixed by the glass dimensions, such that computer control is not possible. It seems necessary that a hardware edge detection device should in orporate some capability for varying the delay, so that various formations of pixel neighbourhoods can be accessed. However, it seems likely that a logical network could be designed to select some subset of glass delay lines for this purpose. The cost for the complete set of delays needed would likely be quite high.

Charge-Coupled Devices (CCDs)

Tompsett and Zimany [46] describe the use of CCDs for delaying analog signals. They show that these MOS devices can operate with effective delays in the range from 100 millisecs to 1 microsec, maintaining fidelity at

frequencies up to several MHz, and having wide dynamic ranges. They illustrate TV images, seemingly non-degraded, which have been delayed 16 millisecs.

Symbolically, the CCD can be thought of as a 'bucket brigade', such that a charge packet (representing the current TV signal voltage) is stepped through the CCD at the command of a timimg pulse. It is then easily understood that the delay obtained through the device is a function of the frequency of this timing pulse. With the appropriate delay, the signals representing one complete TV raster scan line will be within the CCD at some specific instant of time. Furthermore, the signal voltages arriving at the input of the CCD will appear at the output (delayed appropriately) without significant degradation. It is expected that the CCD will be commercially available by August 1973 [44]. Nothing definite can be said concerning cost as yet, but it is reasonable that advancing MOS technology, and large scale production, should make the CCD at least as competitive as the glass delay lines.

Selection Of Delay Devices

One must decide the relative importance of: availability, cost, performance, and flexibility, in order to decide which type of delay device to use in any projected hardware system. It will be assumed that the CCD is the major delay line used, and that other required fixed, short delay devices are of the inexpensive-glass type.

II. A Hardware Edge Detection System

This section will propose the development of a hardware edge detection system whose method of edge detection is dynamically changeable. That is, any subset of a set of hardware-implemented edge algorithms may be operating at any time during the sampling and detecting phase.

The intent of the following description is the encouragement of thought and discussion on the merits, methods, and components for 'hardening' the edge detection process. Consistent with this, the diagrams illustrating the system (Figs. 25,27,28) represent a flow-chart approach rather than the layout of actual hardware.

A Functional Description

Fig. 25 shows a functional block diagram of a hardware edge detecting system. The TV video signal, Stv, is sensed by the delay network (Fig. 27) and is available at various ports of the network as a function of time. A subset of these ports provide delayed signals, SSSd, to each of the edge detecting modules. Using their threshold input from the decoder, each of the modules perform their own unique edge detection operation on the input, producing an edge signal, Sei. These edge indications are summed by a voltage adder. Parenthetically, threshold adjustment could force only one

edge detector to be active at any one time. The Digitizer-Control Unit (DCU) samples the output of the adder, Ssum, once per TV raster line, quantizes it, and passes an edge merit value, EM, to the computer. The computer performs a threshold operation on EM according to how many edge detection modules were active during the operation. The resulting edgels are packed into computer words, and at the end of a TV image column, storage on disc commences. During the relaxation period between TV frames, there is sufficient time to analyze previous results, and reset thresholds for the edge modules as a consequence of that analysis. These new thresholds are converted to useable voltages by a D/A converter, and are dealt to individual modules by a decoder. It is possible to adjust the delay network at this time. However, this would probably not be productive using the configuration of the delay network shown in Fig. 27, except for experimental purposes.

In the foregoing description it was mentioned that the sampling rate was <u>once</u> per TV raster line. This assumes the presence of an analog-to-digital converter using the (slow) successive-approximation technique. However, the computer may only require a <u>binary decision</u> from the DCU regarding the existence of an edgel. If such were the case, then a much faster, simpler, and more inexpensive A/D converter may be utilized: a high speed differential comparator (e.g., Fairchild UA710 -- 100 nanosecs). The output of this type of comparator can be pre-set to voltages

representing logical '1', and '0' for the computer concerned. Thus, the sampling rate would be essentially limited by the computer's cycle time. However, other factors (not the least of which is programming complexity) further restrict this rate to such an extent that sampling more than once per line seems fruitless for the present.

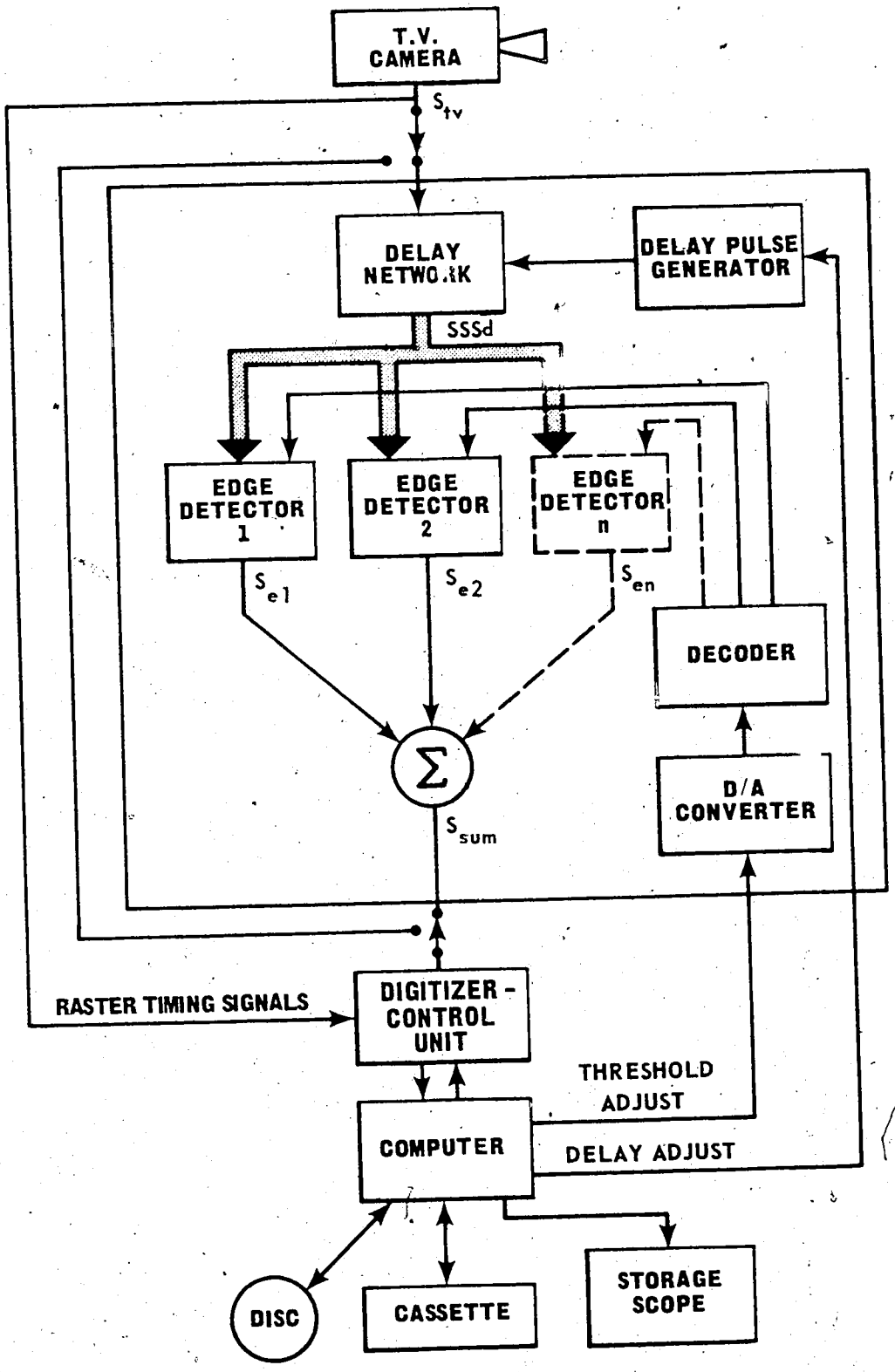


Fig. 25. A hardware edge detection system.

The Delay Network

Fig. 26 shows a matrix representation of a small portion of the TV image. The values in that matrix represent the order in which the corresponding signals arrive at the delay network. These numbers are repeated in the delay

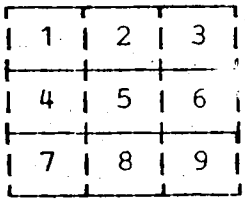
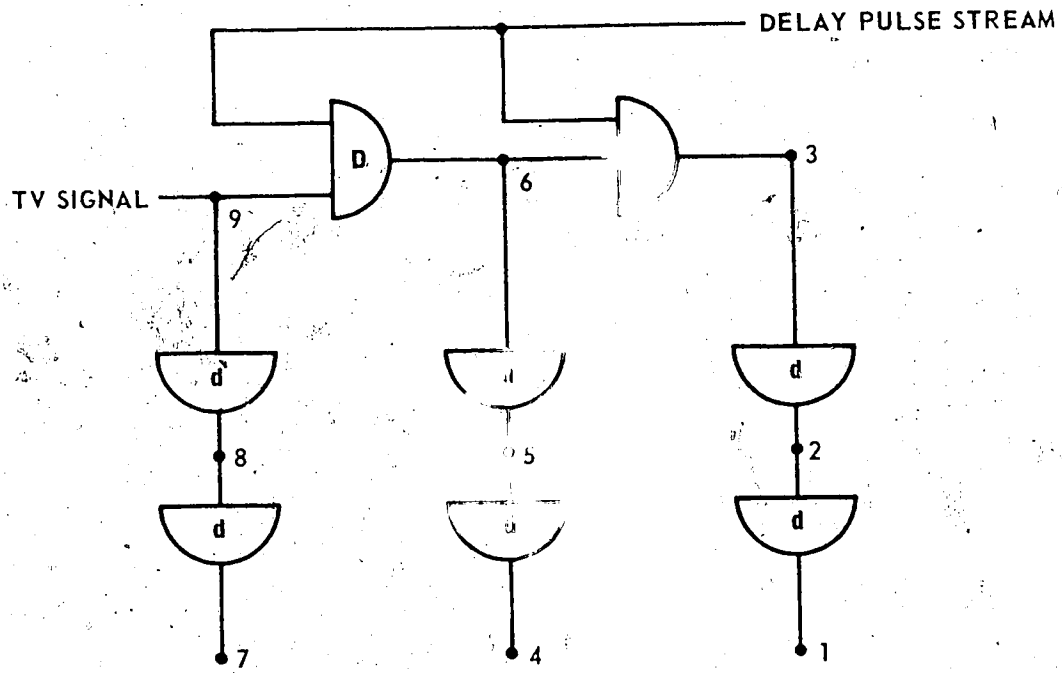


Fig. 26. Sub-array of a picture matrix

network diagram shown at Fig. 27. This particular network is able to provide these nine signals simultaneously to the edge detection modules. The delay network is composed of variable delay CCDs, and fixed-delay glass delay lines. The large delays, D, can be assumed to delay signals 63.5 microsecs (i.e., one TV line), and the small delays, d (remembering the inter-lace problem of a TV raster scan) represent approximately 240 nanosecs. A simple (but more expensive) addition to the network would allow the provision of values from a 4 x 4 matrix, so that a greater variety of edge detecting algorithms could be mechanized.



D – VARIABLE DELAY: APPROX. 63.5 MICROSECS

d – FIXED DELAY: APPROX. 240 NANOSECS

n – SEQUENTIAL PIXEL INDEX NUMBER

NOTES

The delay devices above (i.e., D and d) are, in fact, analog storage devices operating in a fashion similar to a binary shift register:

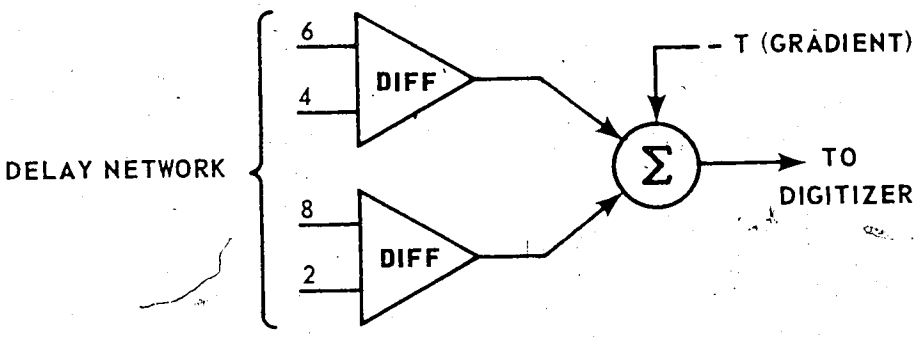
Device D effectively contains within its bounds at any instant a TV signal equivalent in time to one TV raster line.

Device d operates like D, except that the TV signal contained is equivalent to one picture resolution element of the TV line.

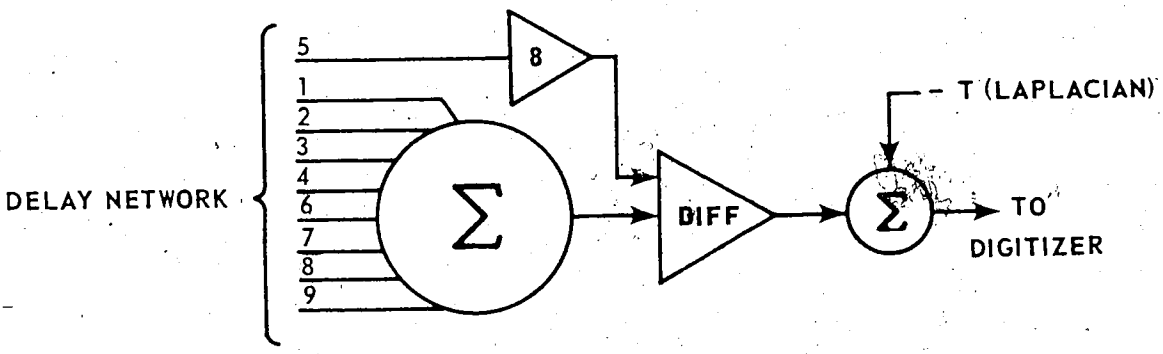
Fig. 27. The delay network.

The Edge Detecting Modules

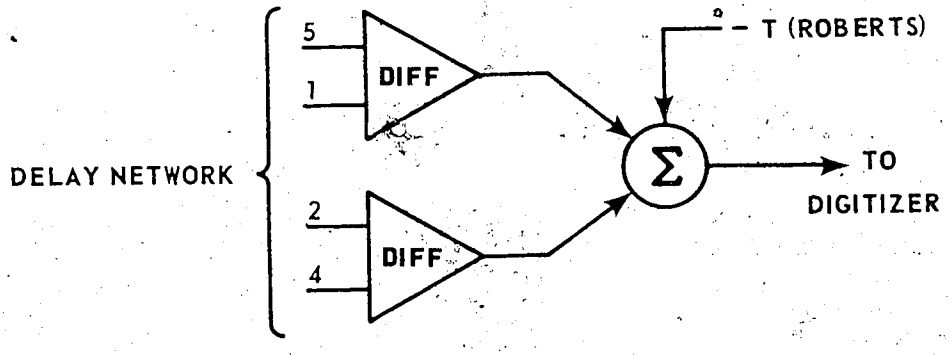
Fig. 28 shows three edge detecting modules which could be constructed: the Gradient, the LaPlacian, and the Roberts. For economy, one module could be designed to handle all three algorithms. Manual switching among the three would then become necessary. Such a design would hinder the possibility of having all three simultaneously active. Further, its more complicated design would offset some of the savings realized by having fewer total components. Thus it is felt that each operator should be constructed as a separate plug-in unit.



a. THE GRADIENT MODULE



b. THE LAPLACIAN MODULE



c. THE ROBERTS MODULE

Fig. 28. Three edge detection modules.

Applications

With an operating hardware system as just detailed, it would be possible to:

a. Investigate the effect of each edge detector on a 'live' image.

b. Examine the interactive effect of several operators working simultaneously.

c. Efficiently determine the class of images for which the various operators are optimal.

d. Investigate dynamic thresholding techniques.

e. Develop edge tracing algorithms within the framework of the exhaustive scanning hardware.

f. Develop new composite software edge detectors as a result of experiments mentioned in sub-paragraph b. above.

Conclusion

This chapter has outlined the development of a hardware edge detecting system. Its capabilities include: speed, flexibility, and experimental interest. Such a system could ease the detection of a wide variety of edge types, since a module could be designed for any specific type of edge expected. In robot applications, the transfer of software edge detection routines to hardware implementation would conserve core memory for other recognition routines. The fact that several hardware edge detectors can effectively 'vote' on the existence of an edge implies that edges can be located with more confidence, and therefore objects can be recognized faster and more accurately. The transfer of this edge operation to hardware does not mean loss of computer control over the process since control of the threshold of all edge detecting modules (and thus their total output) remains soft.

Chapter 6

GENERAL CONCLUSIONS

This thesis has surveyed twelve edge detection techniques with respect to their action on 'real' images. A novel edge detecting algorithm was formulated and developed in Chapter 3, based on the degree of disorder of picture elements within a picture segment. Functional tests of six operators were conducted on real pictures, and particular conclusions were drawn. A hardware edge detection system was broadly outlined in Chapter 5. It now remains to summarize the findings of the analytical examination and the functional tests. The conclusions are abstracted in Table 11.

I. Analytical Evaluation

Remarks about specific operators can be found throughout Chapter 2; however, as the research for this thesis progressed, it became obvious that analytical evaluation of edge detecting methods was of limited value, without knowledge of the precise image on which the detectors were to operate, and without generally accepted standards for evaluation. The only 'hard' conclusions that

could honestly be made were those involving speed, isotropy, and storage costs. It became necessary to 'guess' how good an operator was for characteristics such as noise sensitivity, flexibility, and edge type response.

Analytical Summary

As one might expect, no one operator stood out well above the others. If one wishes a high speed operator, and expects his images to predominate in generally narrow step-type edges, then the Roberts operator is suitable. The LaPlacian is ideal for thin ridges, or highlights, and the Gradient responds positively to wide, steep [illegible]. For ideal step edges the Herskovits operator is indicated. The statistical methods should operate well in the presence of noise. No real conclusions can be drawn about the quasi-parallel methods except that edge detection with them cannot be made context-sensitive, since all edges are detected simultaneously. However, one can say that true parallel processing will result in rapid response (up to several magnitudes, in fact). In general, the analytical evaluation indicated that functional testing was indeed necessary.

II. Functional Evaluation

Based on an analytical examination, six representative edge detectors were implemented as computer programs on the PDP-9 computer. One of these, the Clustering operator, was formulated and developed in Chapter 3.

All six operators were tested on two images: a reference picture, REFPIC, containing a variety of step edges at various orientations; and a 'real' image, FACE, containing a mixture of edge types and orientations. Portions of REFPIC were singled out for specific examination. These were: a diagonal segment, DIAG; a quadrant with alternating black and white segments, CROSS; a narrow right-angled corner, CORNER; and a circle, CIRCLE.

The results of the operations on REFPIC were presented visually and numerically. Assessments were made of the six detectors using storage scope photographs of the various edge transforms as evidence. These assessments were mostly confirmed by the numerical evaluation program MERED. Since this program was developed and implemented during this research it should be regarded as only a first attempt at the 'blind' evaluation of an edge matrix. However, if one examines the output of MERED (i.e., the Fault Tables) without recourse to the photographs, the conclusions are significantly similar to those obtained from the visual examination. Therefore, at least one can say that MERED is sensitive to the same faults as the human eye. Since it was

postulated in Chapter 4 that the eye itself is not necessarily correct in its evaluation of edge matrices, it follows that MERED may be faulty also. Thus, much more work needs to be done along the lines of edge matrix evaluation. The following table and paragraph may give some insight into the problems and possibilities awaiting the prospective investigator.

Table 11. Summary of Evaluation Findings

FACTOR	SUBJECTIVE/ESTIMATED						ANALYTICAL/ACTUAL					
RATED	ROB	GRD	LAP	SML	HER	CLU	ROB	GRD	LAP	SML	HER	CLU
SPEED	17	17	23	111	59	242	80	167	255	617	401	874
BLURRING	F	E	B	A	D	C	F	E	A	D	B	C
GAPS	A	B	E	F	D	C	A	B	F	C	E	D
NOISINESS	D	B	C	F	E	A	A	C	F	E	D	B
MATRIX MERIT	D	B	C	F	E	A	A	C	F	E	D	B

Notes:
a. Names of operators have been abbreviated.
b. Ratings are alphabetic, and merely indicate order of precedence, i.e., 'C' does not necessarily imply an average operator.
c. Edge detector speeds are in CTUs per local operation (estimated), and in micro-secs per local operation (actual). The LaPlacian rated here is the slower version, i.e., equation (7).
d. Ratings were abstracted from Tables 1 and 2.
e. Although certainly desireable, the evaluation of these detectors by more than one individual was not possible due to time constraints.

A Perplexing Note On Table 11

The time sequence of the following events is important to the substance of this paragraph. During the final stages of thesis preparation, the ratings in Table 11 were abstracted from the photographs and fault tables. The subjective ratings for Blurring, Gaps, and Noisiness were completed first; then later, the analytical results for these factors were produced. Still later, the complete set of photographs was examined to determine subjectively (without regard to the previous ratings made above) the best overall edge operator (aside from speed). This resulted in the ratings under the subjective column for the Matrix Merit factor. Finally, the analytical ratings for Matrix Merit were produced. It is startling to examine the last two lines of Table 11, and find them identical. There being no typographical error, one can perhaps assume that the author (although trying to be objective) was strongly sensitive to visual noise, and so was his program MERED (even though the analytical Matrix Merit rating was the mathematical average of the three other factors). Further, it is interesting to note the probabilistic correspondence between the subjective ratings for Matrix Merit and those under the analytical column set. If the analytical ratings were randomly assigned, one could expect an average difference in alphabetic rating between the two column sets of 2.5 per edge operator. The actual average difference is only 1.666..., seemingly, a significant departure from

randomness, when one considers that 4 of the 6 ratings had a difference of only 1. Consideration of subjective versus analytical ratings for only GAPS and BLURRING results in an actual average difference of 1.00. This last fact is additional evidence that successful results may be achievable in the absolute evaluation of edge matrices by algorithmic measurement.

Functional Summary

Visually, the Gradient and the Clustering operator gave the best results, and the Statistical Modified LaPlacian (SML) gave the worst. However, the SML gave rise to some interesting questions with its surprisingly sharp vertical and horizontal lines, embedded in a sea of noise. The reasons for this are not clear, and should be examined further. As mentioned previously, the numerical ratings confirmed the visual examination. The difference in speeds of the various operators ranged over a magnitude, with the Roberts being the fastest, and the Clustering operator the slowest. These differences were due to the number of pixels accessed per local operation, the number of times each pixel was accessed, the complexity of the operation performed, and perhaps more importantly, differences in the efficiency of array indexing among the various operators.

Although a statement of the form: "Operator X is the best." would seem to be appropriate in this concluding chapter, it cannot be given. The reason for this is the lack

of generally accepted standards of evaluation. Before continuing the development of 'new and better' edge detecting algorithms, it is mandatory that edge matrix - evaluation methods be formulated and standardized.

III. Hardened Edge Detection

A proposal and outline for a hardware edge detection system was developed in Chapter 5. The realization of this system should be technically easy, relatively inexpensive, and could be integrable within existing picture processing systems, with a minimum of fuss. Experiments with such a system should accelerate the development of appropriate detectors, and might even suggest methods for their evaluation. The fact that several edge detectors can be used on an image simultaneously implies that the total edge detecting process could become reasonably independent of the class of input images. 'Good' results could then be expected from widely varying picture types. The ability to control the thresholds of the edge detecting modules from within the computer means that an image can be re-scanned several times with settings based on the results of previous scans. Thus one should be able to 'search' for the most satisfactory edge transform.

IV. Closing remarks

The termination of research on this thesis has not resulted in a feeling of 'job completed!'. Many avenues have been unexplored, and too many techniques have been left untried. More time is needed to examine, for instance: dynamic thresholding techniques, contour following and gap filling algorithms, histogram equalization, and Hadamard transformations. These few examples are relevant and important to the task of edge detection.

Hubel and Weisel [19] state: "one of the most fundamental operations performed by the eye is that of detecting changes in illumination". These changes generate the concept of edge within the human mind. Since this edge-locating operation is so fundamental to human existence, one must expect its effective automation to be accomplished only with a great deal of thought, imagination, and effort.

BIBLIOGRAPHY

(1) The British Computer Society, Second Joint International Conference on Artificial Intelligence (London, June 1971).

(2) G. Cheng et al., Eds., Pictorial Pattern Recognition. Washington, D.C.: Thompson, 1968.

(3) IEEE Conference Record of the Symposium on Feature Extraction and Selection in Pattern Recognition (Argonne, Ill., October 1970).

(4) B. Lipkin and A. Rosenfeld, Eds., Picture Processing and Psychopictorics. New York: Academic Press, 1970.

(5) J. T. Tippett et al, Eds., Optical and Electro-optical Information Processing. Cambridge, Mass.: MIT Press, 1965.

(6) A. Grasselli, Ed., Automatic Interpretation and Classification of Images. New York: Academic Press, 1969.

REFERENCES

[1] H.C. Andrews et al,"Image Processing by Digital Computer," IEEE Spectrum, vol. 9, Jul 1972, pp. 20-32.

[2] W.S. Boyle and G.E. Smith,"Charge coupled semiconductors devices," Bell Systems Tech. J. April 1970, p. 587.

[3] C.K. Chow and T. Kaneko,"Boundary detection of radiographic images by a threshold method," IFIP Congress 71. North Holland, Amsterdam, Aug. 1971, Booklet TA-7, pp. 130-134.

[4] Corning Memory Products,"Product bulletin, MCA-5.04, Video Delay Line," Corning Glass Works, 3900 Electronics Drive, Raleigh, North Carolina.

[5] Henry S. Craumer,Personal Communication, Corning Glass Works, Raleigh, North Carolina.

[6] M.F. Dacy and T.H. Tung,"The Identification of Randomness in Point Patterns," J. Reg. Sci., vol. 4, 1962, pp. 55.

[7] R.O. Duda and P.E. Hart,"Use of the Hough transformation to detect lines and curves in pictures," Comm. ACM. Vol. 15, Jan. 1972, pp. 11-15.

[8] Editorial,"Special report," ELECTRONICS, Feb 28, 1972, p. 34.

[9] G. Falk,"Scene analysis based on imperfect edge data," op. cit. (1), pp. 8-16.

[10] J.A. Feldman et al,"The Stanford hand-eye project," Proceedings of the First international Joint Conference on Artificial Intelligence, May 1969, pp. 521-526.

[11] G. Forsen,"Processing digital data with an automaton eye," op. cit. (2), pp. 471-502.

[12] W.D. Fryer and G.E. Richmond,"Two dimensional spatial filtering and computers," Proc. Nat. Elect. Conf., vol. 18, 1962, pp.529-535.

[13] A.K. Griffith,"Mathematical models for automatic line detection," op. cit. (1), pp. 17-26.

[14] E.L. Hall et al,"Measurement selection techniques applied to digital images," op. cit. (3), pp. 78-89.

[15] Joseph K. Hawkins,"Parallel Electro-Optical Picture Processing," op. cit. (2), pp. 373-385.

[16] A. Herskovits,"On boundary detection," MIT Project MAC, AI Memo 183, 1970.

[17] W.S. Holmes et al.,"Design of a photo-interpretation automaton," 1962 Fall Joint Computer Conf., AFIPS Conf. Proc., pp. 27-35.

[18] T.S. Huang et al,"Image processing," Proc. IEEE, vol. 59, Nov. 1971, pp. 1586-1609.

[19] D.H. Hubel and T.N. Wiesel,"Receptive fields of single neurons in the cat's striate cortex," J. Physiol., vol. 148,1959, pp. 574-591.

[20] M.H. Hueckel,"An operator which locates edges in digitized pictures," Stanford Artificial Intelligence Project Memo AIM-105. Oct. 1969.

[21] E.G. Johnston,"The PAX picture processing system," op. cit. (4), pp. 427-512.

[22] G.S. Kino and John Shaw,"Acoustic surface waves," Scientific American, vol. 227, Oct 1972, pp. 50-68.

[23] L.S.G. Kovasnay and H.M. Joseph,"Image processing," Proc. IRE, vol. 43, May 1955, pp. 560-570.

[24] R.S. Ledley and F.H. Ruddle,"Chromosome analysis by computer," Scientific American, vol. 214, April 1966, pp. 40-46.

[25] J.Y. Lettvin et al,"What the frog's eye tells the frog's brain," Proc. IRE. Vol. 47, 1959, pp. 1940-1951.

[26] M. Levine,"Feature extraction: a survey," Proc. IEEE, vol. 57, Aug. 1969, pp. 1391-1407.

[27] A.W. Lohman and D.P. Pairs,"Computer generated spatial filters for coherent optical data processing," Appl. Opt., vol. 7, April 1968, pp. 651-655.

[28] J. Mccarthy,"Plans for the Stanford artificial intelligence project," Stanford University Report, April 1965.

[29] B.H. Mccormick,"The Illinois pattern recognition computer - ILLIAC III," IEEE Trans. Electronic Computers, vol. EC-12, Dec 1963, pp. 791-813.

[30] M. Minsky and S.L. Papert,"Research on intelligent automata," Status Report II, Project MAC, MIT, Sept. 1967.

[31] J.L. Muerle and D.C. Allen,"Experimental evaluation of techniques for automatic segmentation of objects in a complex scene," op. cit. (2), pp. 3-13.

[32] John L. Pfaltz et al,"Local and global picture processing by computer," op. cit. (2), pp. 353-371.

[33] K.K. Pingle and J.M. Tenenbaum,"An accommodating edge follower," op. cit. (1), pp. 1-7.

[34] W.K. Pratt et al,"Hadamard Transform Image Coding," Proc. IEEE, vol. 57, Jan 1969, pp. 58-68.

[35] K. Preston, Jr.,"Use of the Fourier transformable properties of lenses for signal spectrum analysis," op. cit. (5), pp. 59-68.

[36] L.G. Roberts,"Machine perception of three dimensional solids," op. cit. (5), pp. 159-197.

[37] P.G. Roetling et al,"Theoretical prediction of image quality," J. Opt. Soc. Amer., vol. 58, Mar. 1968, pp. 342-346.

[38] A. Rosenfeld, Picture Processing by Computer. New York:Academic Press,1969.

[39] A. Rosenfeld et al,"Edge and Curve Detection for Texture Discrimination," op. cit. (4), pp. 381-393.

[40] A. Rosenfeld and M. Thurston,"Edge and curve detection for visual scene analysis," IEEE Trans. Comp., vol. C-20, May 1971, pp. 562-569.

[41] A. Rosenfeld and E.B. Troy,"Visual texture analysis," op. cit. (3), pp. 115-124.

[42] D. Rutovitz,"Data structures for operations on digital images," op. cit. (2), pp. 105-133.

[43] L.K. Schubert, Private Communication, Jun. 1972.

[44] J. Sevick, Private Communication, Bell Laboratories, 600 Mountain Road, Murray Hill, N.J.

[45] Y. Shirai and S Tsuji,"Extraction of the line drawings of 3-dimensional objects by sequential illumination from several directions," op. cit. (1), pp. 71-79.

[46] M.F. Tompsett and E.J. Zimany, Jr.,"Use of charge-coupled devices for delaying analog signals," to be published in IEEE J. Solid State Circuits, 1973.

[47] E.A. Trabka and P.G. Roetling,"Image transformations for pattern recognition using incoherent illumination and bipolar aperture masks," J. Opt. Soc. Amer., vol. 54, Oct. 1964, pp. 1242-1252.

[48] P.M. Will and K.S. Pennington,"Grid coding: a preprocessing technique for robot and machine vision," op. cit. (1), pp. 66-70.

VITA

NAME: Melvin Wesley Smith

PLACE OF BIRTH: Edmonton, Canada

YEAR OF BIRTH: 1932

POST-SECONDARY EDUCATION AND DEGREES:

University of Alberta
Edmonton, Canada
1969-1971 B.Sc.

RELATED WORK EXPERIENCE:

Officer-Canadian Armed Forces
Department of National Defence
1955-

Computer Systems Evaluation and Testing
Department of National Defence
1973-

PUBLICATIONS:

(1) "A System for Processing Digital Pictures", Proc. 3rd Man-Computer Communications Seminar, National Research Council of Canada, May 1973, pp. 3.1-3.9.