REPORT ON THE FAST BOUNDARY DETECTION ALGORITHM OF WERNER FREI AND CHUNG-CHING CHEN

by

DANIEL BENJAMIN SCHOWENGERDT

B. S., Kansas State University, 1979

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1983

Approved by:

Kasi Ahmed
Major Professor

TABLE OF CONTENTS

THIS BOOK CONTAINS NUMEROUS PAGES WITH THE ORIGINAL PRINTING BEING SKEWED DIFFERENTLY FROM THE TOP OF THE PAGE TO THE BOTTOM.

THIS IS AS RECEIVED FROM THE CUSTOMER.

ILLEGIBLE DOCUMENT

THE FOLLOWING DOCUMENT(S) IS OF POOR LEGIBILITY IN THE ORIGINAL

THIS IS THE BEST COPY AVAILABLE

THIS BOOK CONTAINS NUMEROUS PAGES WITH DIAGRAMS THAT ARE CROOKED COMPARED TO THE REST OF THE INFORMATION ON THE PAGE.

THIS IS AS RECEIVED FROM CUSTOMER.

LIST OF FIGURES

TO

1) The patience and tolerance of my committee members,

2) The enjoyment of research and hopefully the death of my procrastination,

3) And to a Friend.

ACKNOWLEDGMENTS

The author would like to thank Sandia Laboratories, Albuquerque, N.M. for providing financial support and the K.S.U. Electrical Engineering Department for the computing facilities used in connection with this report.

The author would also like to acknowledge the privilege of being taught by and working with Dr. D.H. Lenhert, Dr. M.S.P. Lucas and Dr. N. Ahmed as well as other faculty members of the K.S.U. Electrical Engineering Department. Thanks is also extended to persons who have served as committee members, my major professor Dr. N. Ahmed, Dr. D.H.Lenhert, Dr. D.W. Curtis, and Dr. D. Hummels.

CHAPTER 1

INTRODUCTION

The subject of this report is a paper by Werner Frei and Chung-Ching Chen entitled "Fast Boundary Detection: A Generalization and a New Algorithm" [1]. This paper introduces a mathematical concept of boundary detection in digital images and presents results of the related implementation. This report consists of an introduction to the subject of boundary or edge detection, a description of their concept, and independent implementation results.

One of the motivations for research in edge detection in digital images is that computers and humans can easily record images but do not as easily analyze and understand the information in the images. Edge detection is one method for increasing the ability to understand an image by highlighting boundaries or edges found in the image. For example, edge detection may enhance the ability to identify objects in an image by making distinct outlines of the objects.

CHAPTER 2

SOME FUNDAMENTALS

To define the problem of edge detection there must first be an understanding of digital images, as recorded by a computer. A simple way to understand digital images is to imagine a black and white photograph that has been carefully cut into, for example, 256 horizontal strips of the same width. Then the picture is cut into 256 same width columns. Each of the resulting squares, called picture elements or pixels for short, is then compared to a scale of grey tones which vary in brightness from black at one extreme, to white at the other. There are 256 levels of grey tones, each assigned a number. For example, 0 may be assigned to black and the numbers increase and tones brighten until white is reached with the assigned value of 255. When the average grey tone of the pixel matches the grey tone of the scale the pixel is assigned that number. A computer thus records an image by keeping track of the row and column that a pixel is in and the corresponding grey scale value. To a computer then, an image consists of a set of numbers which represents to it the tones of grey that the human eye perceives.

A needed additional understanding is what is meant by the term edge. A general definition would be that an edge is where two regions of different homogeneous content meet. For example, in a color photo of the flag of the United States, an edge would be where a white stripe

meets a red stripe. If the same photo was in black and white the edge would no longer be between regions of different color but between regions of different grey tones. This type of edge between regions of different grey tones, is the type of edge of interest in the paper by Frei and Chen [1]. In a digital image, where numbers represent grey tones, edges are thus where these grey scale numbers change between one group of pixels and another.

These changes in pixel grey scale values are the basis for two opposite methods of edge detection. The first method is to start with an image and find small groups of pixels that have similar grey scale levels. Then adjacent pixels are added to these groups until the adjacent pixels have different enough grey scale values to judge them as being from a different group. Thus small "seed" regions in a picture are "grown" until they meet other regions and edges are formed. In this method the emphasis is on looking for pixels with similar grey scale values.

The second method is to start with an image and find adjacent pixels that have different grey scale values. On the basis of the size of the difference, a decision is made as to whether the pixels are part of an edge. The algorithm of Frei and Chen is one of several algorithms that use this method for finding edges.

Unfortunately the problem of edge detection is not as simple as just finding the difference between the grey scale values of two adjacent pixels. One complicating factor is noise which can come from the lighting of the scene being recorded, from the photosensitive parts of the camera and/or the digitizing of the grey tones. Noise

has the effect of randomly and sometimes not randomly changing the grey scale value of pixels. This tends to blur digital images and make those pixel value changes at edges erroneously larger or smaller. Another complicating factor caused by noise, the limitations of camera hardware and by the reality of true edges is that sometimes edges are not recorded completely by just two pixels. For example, consider three pixels in one row. Assume these pixels are a digital image of an edge and the pixel values, in order, are 130, 120 and 110. The difference between adjacent pixel values is 10. If the criterion for there being an edge was a difference of 15, no edge is detected. If, however, the two outer pixel values are considered, the difference is 20 and an edge is detected as it should be. If noise were added to this example so that the values changed to 127, 119 and 113, then even with consideration of the outer two values, an edge would not be detected. To correct for this problem of noise, pixel values adjacent to the 127 and 113 valued pixels could be considered and, for example, averaged. The resulting averages could then be differenced and an edge determined. This averaging, for example, may have the advantage of less errors caused by noise, but by involving large number of pixels increases computation time and increases the possibility of not detecting two edges separated by only one pixel width.

Often algorithms vary from user to user because of new insight as well as different applications. However, most of them follow the same pattern of first processing an image with a simple edge detector. The edge segments found are then linked together and thinned to obtain the best results. Frei and Chen mention several published algorithms

including ones by Roberts [2], Kirsh [1], Sobel [4], Prewiit [5] and Robinson [6]. They then mention that these algorithms revealed similarities which led to the development of their algorithm which is introduced in the next section.

CHAPTER 3

BASIC CONCEPTS AND DEFINITIONS

The problem of determining if a pixel element is an edge element, as defined by Frei and Chen, but using terminology from the first section of this report, is as follows: given a set of n^2 adjacent scale values from a small portion of an image, called a subarea, determine whether the subarea contains an edge pixel between two regions of different homogeneous grey scale values. Frei and Chen also add that it may be of interest to determine if the subarea contains a line, a line being two edges so close together that they are difficult to distinguish as being separate.

To further define the problem, Frei and Chen define three ideal images that can occur in a subarea: the ideal edge, the ideal line and the ideal point.

The image of an "ideal edge" occurs when the edge is straight and passes through the center of the subarea [see Fig. 1(b)]. When this subarea is digitized the result will be pixel grey scale values of $\bar{b}_1$ on one side of the edge and pixel grey scale values of $\bar{b}_2$ on the other. With the convention $\bar{b}, >\bar{b}$ then a direction θ_e for the edge can be uniquely determined with respect to some arbitrary fixed direction. The edge in this subarea is then characterized by its "magnitude" = $|\bar{b}_1 - \bar{b}|$ and direction $\theta_e, 0 \leq \theta_e < 2\pi$.

The image of an "ideal line" occurs when a straight stripe passes

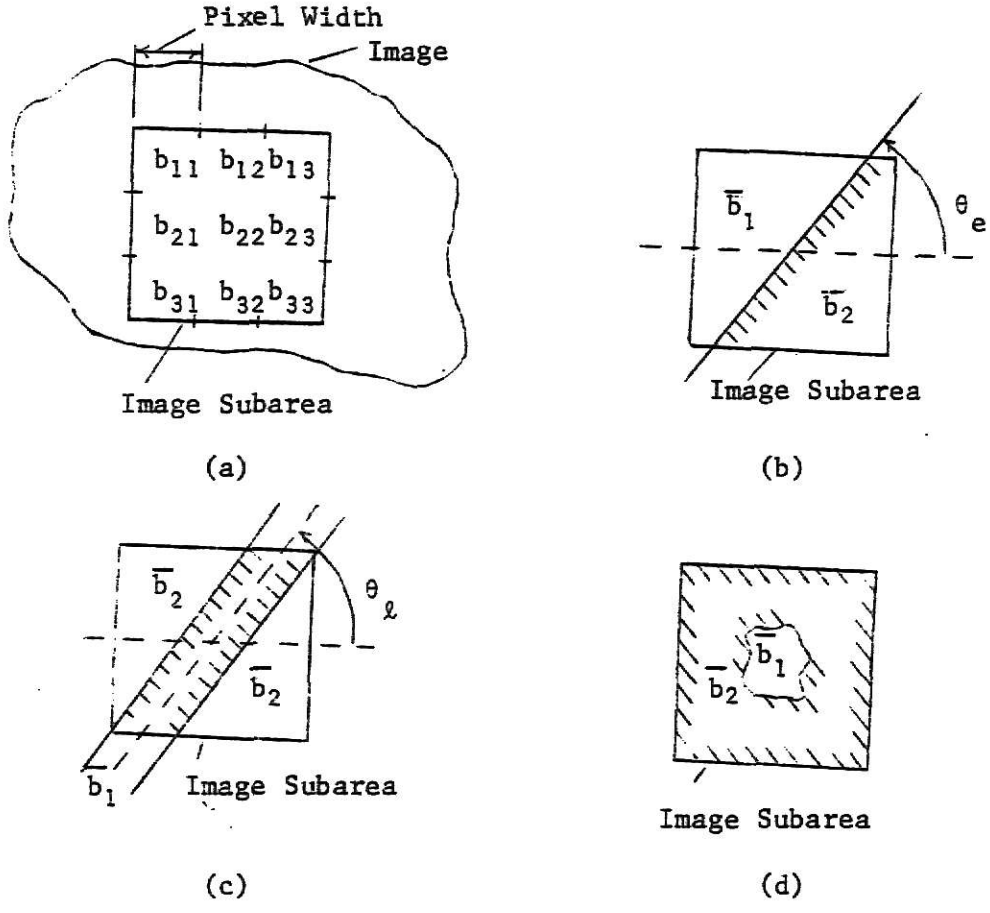


Figure 1. (a) Definition of image subarea with components of subarea vector shown for the 3x3 pixel case,
(b) Definition of ideal edge subarea,
(c) Definition of ideal line subarea,
(d) Definition of ideal point subarea.

through the center of the subarea which has a width of about one pixel width [Fig. 1(c)]. When this subarea is digitized the pixels on the line will have value $\bar{b}_1$ and those on either side $\bar{b}_2$. As with the edge, a direction θ_ℓ can be determined but its range is only $0 \leq \theta_\ell < \pi$. Again, the "magnitude" = $|\bar{b}_1 - \bar{b}_2|$ and angle θ_ℓ, with the addition of polarity $(\bar{b}_1 - \bar{b}_2)$, characterize the line.

Finally, the image of an "ideal point" occurs when there is a region in the center of the subarea whose area is about equal to the area of one pixel [Fig. 1(c)]. When this image is digitized the center pixel will have a value $\bar{b}_1$, surrounded by pixels with value $\bar{b}_2$. This point is characterized by its "magnitude" = $|\bar{b}_1 - \bar{b}_2|$ and its polarity $(\bar{b}_1 - \bar{b}_2)$. More definitions which apply to the problem of edge detection concern the mathematics involved. This mathematics is simply vector and matrix mathematics. The rows and columns of a matrix correspond directly to the rows and columns of pixel grey scale values of a digital image. Thus the digitized pixel values b_{ij} (b: represents pixel grey scale value, i: pixel row, j: pixel column) can be represented by a matrix. For example, if the subarea was a set of n^2 pixels and n=3, the pixel values could be represented by the matrix B:

$$B = \begin{matrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{matrix} \qquad \text{[See Fig. 1(a)]}$$

An alternate method is to arrange the pixel values into a vector b:

$$b = [b_{11}, b_{12}, b_{13}, b_{21}, b_{22}, b_{23}, b_{31}, b_{32}, b_{33}].$$

With this notation the mathematical operation of inner or dot product (.,.) between matrices is defined as

$$(B,C) = \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} c_{ij}.$$

For vectors, the inner product is a simpler operation:

$$(b,c) = \sum_{k=1}^{n^2} b_k c_k.$$

This is simpler because the pixel values now have one index corresponding to their order in the vector instead of two indices, one for the row and one for the column.

These definitions are not unique to the algorithm of Frei and Chen. They apply, with perhaps some slight modifications to all the algorithms referenced, as well as others which fall in a general category of enhancement/threshold edge detector algorithms. Enhancement refers to the process of determining the quantity of a certain quality in a subarea of a digitized image. Threshold refers to the decision then made based on the quantity. If the quantity does not exceed the threshold level, the subarea is judged to not possess the desired quality. If it does exceed the threshold, the subarea is judged to possess that quality.

Chapter 4

CATEGORIES OF ENHANCEMENT/THRESHOLD EDGE DETECTION METHODS

Enhancement/threshold edge detector algorithms fall into two categories [1], [7], based on their method of enhancement. The difference is in the types, numbers, and combination of results of operators designed to enhance certain qualities of a subarea.

One category uses differential operators which perform a discrete differentiation on a subarea to yield a result corresponding to the gradient. This category includes the Roberts [2], Prewitt [5], and Sobel [4,p.271] operators [see Fig. 2]. The basic assumption for this approach is that subareas containing edges will yield a large gradient because of the difference between the grey scale values between regions on either side of the edge. If the edge operators are indicated as matrices W_1 and W_2, and the same size image subarea as matrix B, the magnitude of the gradient is obtained by the equation:

$$A = [\ (B,W_1)^2 + (B,W_2)^2\ ]^{1/2}. \qquad (1)$$

If the scalar value A is greater than some threshold, then the center pixel of the image subarea is considered an edge pixel. If it is less than the threshold then the center pixel is not considered an edge pixel. An equation that is sometimes used as a gradient indicator because it is simpler than equation (1) is:

$$A = |\ (B,W_1)\ | + |\ (B,W_2)\ |. \qquad (2)$$

$$\begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix} \quad \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix}$$

Roberts Operators

$$\begin{vmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{vmatrix} \quad \begin{vmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{vmatrix} .$$

Prewitt Operators

$$\begin{vmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{vmatrix} \quad \begin{vmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{vmatrix}$$

Sobel Operators

$$\begin{vmatrix} 1 & \sqrt{2} & 1 \\ 0 & 0 & 0 \\ -1 & -\sqrt{2} & -1 \end{vmatrix} \quad \begin{vmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{vmatrix}$$

Isotropic Operators

Figure 2.

Gradient Operators for Edge Detection
([2], [5], [4], [1])

There is also an equation for the approximation of the angle of the edge with reference to direction of the operators:

$$\theta = \arctan [(B,W_1) / (B,W_2)].$$

A comparison of the response of these operators, using equations (1) and (2), to an ideal edge being rotated through 45°, shows that there are differences [see Fig. 3]. These and other differences can be seen in an article by Abdou and Pratt [7]. Robinson [6] tabulates the normalized response of these operators to a 45° ideal edge resulting in the table:

Operator [see Fig. 2]	Angle	Amplitude A (equ. 1)
Prewitt	45°	0.943
Sobel	45°	1.067
Isotropic	45°	1.000

The response amplitude of the isotropic operators has been added to Robinson's table. From these responses, Robinson chose the Sobel operators [see Fig. 2] because he feels its greater response to 45° edges compensates for lower visual acuity in diagonal directions. However, Frei and Chen chose the isotropic operators because they are the best of the operators shown in Fig. 2 at being invariant in amplitude response to changes in edge angle.

The other category of enhancement/threshold edge detectors uses template matching operators [see Fig. 4]. These operators are sets of matrices representing discrete approximations to ideal edges of various orientations. These operators include the compass gradient operators introduced by Prewitt [5], the Kirsch operators [3], and

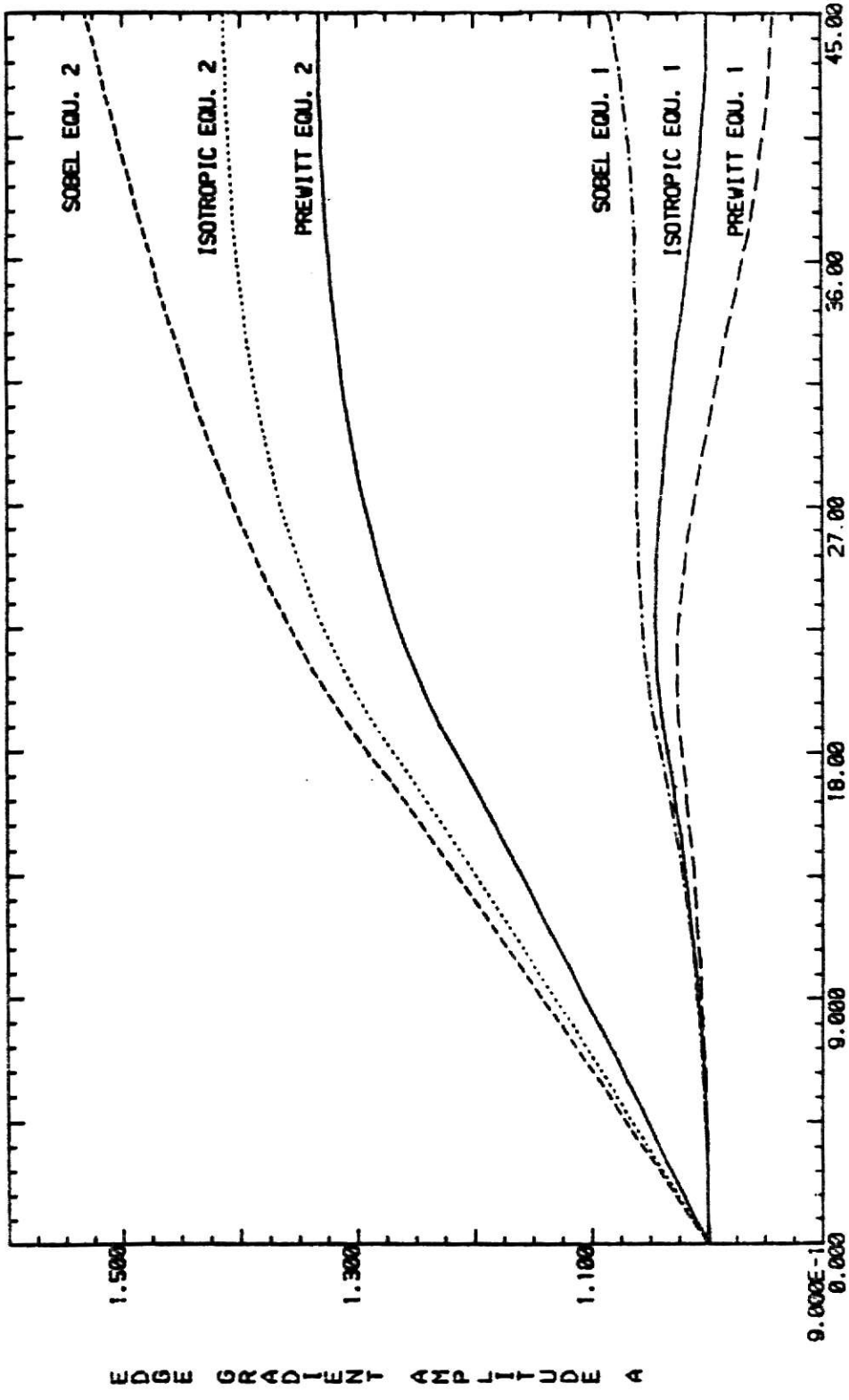


Fig. 3. Edge gradient amplitude response, equ. (1) and (2), as a function of actual edge orientation for the 3 x 3 operators of Fig. 2. Response is normalized to unity for a vertical edge.

Direction of Gradient	Compass Gradient	Kirsch	3-Level Simple	5-Level Simple
North	1 1 1 1 -2 1 -1 -1 -1	5 5 5 -3 0 -3 -3 -3 -3	1 1 1 0 0 0 -1 -1 -1	1 2 1 0 0 0 -1 -2 -1
Northwest	1 1 1 1 -2 -1 1 -1 -1	5 5 -3 5 0 -3 -3 -3 -3	1 1 0 1 0 -1 0 -1 -1	2 1 0 1 0 -1 0 -1 -2
West	1 1 -1 1 -2 -1 1 1 -1	5 -3 -3 5 0 -3 5 -3 -3	1 0 -1 1 0 -1 1 0 -1	1 0 -1 2 0 -2 1 0 -1
Southwest	1 -1 -1 1 -2 -1 1 1 1	-3 -3 -3 5 0 -3 5 5 -3	0 -1 -1 1 0 -1 1 1 0	0 -1 -2 1 0 -1 2 1 0
South	-1 -1 -1 1 -2 1 1 1 1	-3 -3 -3 -3 0 -3 5 5 5	-1 -1 -1 0 0 0 1 1 1	-1 -2 -1 0 0 0 1 2 1
Southeast	-1 -1 1 -1 -2 1 1 1 1	-3 -3 -3 -3 0 5 -3 5 5	-1 -1 0 -1 0 1 0 1 1	-2 -1 0 -1 0 1 0 1 2
East	-1 1 1 -1 -2 1 -1 1 1	-3 -3 5 -3 0 5 -3 -3 5	-1 0 1 -1 0 1 -1 0 1	-1 0 1 -2 0 2 -1 0 1
Northeast	1 1 1 -1 -2 1 -1 -1 1	-3 5 5 -3 0 5 -3 -3 -3	0 1 1 -1 0 1 -1 -1 0	0 1 2 -1 0 1 -2 -1 0

Figure 4. Examples of Template Matching Operators

the 3- and 5-level simple operators [6]. With these operators the inner product of the image matrix B with each template matrix T_i is computed and the largest response is retained. The equation is:

$$A = MAX\ (B,T_i) \quad i = 1,2,..,9$$

If the response magnitude A is greater than some arbitrary threshold then the subarea is considered to contain an edge pixel. If not, then there is no edge pixel. The basic assumption of this category is that if there is an edge in the subarea it will be enhanced by one of the templates which approximate all possible edges. This assumption can also be applied to the detection of lines and points by using templates that are discrete approximations of ideal line and point images. Figure 5 shows some examples of such templates.

$$\begin{vmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{vmatrix} \qquad \begin{vmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{vmatrix}$$

T_1 T_2

$$\begin{vmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{vmatrix} \qquad \begin{vmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{vmatrix}$$

T_3 T_4

(a) Line Templates

$$\begin{vmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{vmatrix}$$

(b) Point Template

Figure 5. Templates used for (a) line detection and (b) point detection.

Chapter 5

THE EDGE DETECTION ALGORITHM OF FREI AND CHEN

In the formulation of their algorithm Frei and Chen used vector mathematics. Since the definition of projection from vector mathematics is: given two vectors B and W, the projection of B on W is the vector [(B,W) / (W,W)] * W. They conclude that all previous enhancement/threshold algorithms are thresholding the magnitude of the projection of the image subarea vector B on the differential or template matching operator vectors. They also draw other conclusions about the relationship between the subarea vector B and the operator vectors. One conclusion they draw is that the image vector B and the operator vectors are only a few of the vectors in a "space". This "space" is defined by vector mathematics as all possible n^2 element vectors. Another conclusion is that the purpose of the operator vectors is to describe a smaller set of vectors or a "subspace" which is part of this space. For example, the subspace described by the gradient operators W_1 and W_2 are all vectors W such that

$$W = c_1 W_1 + c_2 W_2 .$$

Since the thresholded value is the magnitude of projection onto these subspaces the desire of users of these vectors is to design operators that very closely describe all possible image edge vectors.

Again, from an application of vector mathematics, Frei and Chen

realize that this magnitude of projection is not the only quantity between a vector and a subspace that can be thresholded. There is also the angle between the vector and the subspace. Between two vectors B and T the angle is defined as:

$$\theta = \arccos [(B,T) / (||B|| * ||T||)].$$

Frei and Chen use this angle θ to determine how well the image vector B "fits" the ideal edge subspace described by certain operator vectors.

In order to choose these operators, Frei and Chen used vector mathematics to understand the n^2-dimensional space with which they are working. They realize that if there is a set of n^2 vectors $W_1, W_2, .., W_{n^2}$ from this n^2-dimensional space, which have the property

$$c_1W_1 + c_2W_2 + ,.., + c_{n^2}W_{n^2} = 0$$

only if $c_1, c_2, .., c_{n^2} = 0$, then these vectors would form a basis set of vectors for the space. This property called linear independence implies that given any possible vector V in the space, the set would be a basis set because this equation would always be true:

$$V = c_1W_1 + c_2W_2 + ,.., + c_{n^2}W_{n^2}.$$

It is not mathematically difficult to show that if a set of n^2 vectors are orthogonal, that is they have the property

$$(W_i,W_j) = \begin{cases} k_i & i=j \\ 0 & i \neq j \end{cases},$$

then they are also linearly independent.

In order for their angle algorithm to work, Frei and Chen realize

that the operator vectors they work with must be orthogonal. If they are not orthogonal then an inaccurate angle value would result. With this understanding they realize the restrictions they are working under. In the space they are working with, they must choose e orthogonal vectors [$T_1,..,T_e$] which describe an "edge" subspace. This subspace may be the same as those described by vectors of previous algorithms [see Figures 2,4], but now there is a restriction of orthogonality. Once these e "edge" vectors are chosen then there are n^2-e "non-edge" vectors [$T_{n^2-e},..,T_{n^2}$] to choose to complete a basis set for the n^2-dimensional space. Once this is done then the angle between the image subarea vector B and the "edge" subspace is

$$\theta = \arccos \left[\sum_{i=1}^{e} (B,T_i)^2 / \sum_{j=1}^{n^2} (B,T_j)^2 \right]^{1/2}.$$

Since the angle θ quantifies the similarity between the image subarea and the edge subspace, Frei and Chen propose that if the resulting angle θ is less than some threshold, that the image subarea contains an edge pixel.

Realizing that since the set of vectors $T_1,..,T_{n^2}$ are a basis set so that

$$(B,B) = \sum_{i=1}^{n^2} (B,T_i)^2 , \qquad (4)$$

Frei and Chen simplify their algorithm to thresholding the magnitude of this equation:

$$\sum_{i=1}^{e} (B,T_i)^2 \; / \; (B,B). \qquad (5)$$

In comparing this equation (5) to equation (1) and (2) used in previous algorithms, it is clear that the only difference is the use of the denominator. To understand the effects of this denominator consider that (B,B) is the square of the length of the image vector. The equation then becomes one of finding the ratio between the projection of the image vector on the edge subspace and the projection of the image vector on itself. It is as if the image vector were divided into two new vectors, an edge vector and a non-edge vector. A comparison is then made between the size of these vectors and the size of the original vector. If the edge vector is large enough in length, then an edge pixel has been discovered. Previous algorithms have not used this comparison and so may erroneously classify pixels as edge pixels when the length of the non-edge vector may have been as large, if not larger than the length of the edge vector.

A good comparison of the differences with with this new algorithm can be seen in Fig. 6 [1]. Frei and Chen describe this figure as

> "two subarea vectors B_1 and B_2 are shown, projected onto the "edge" and "non-edge" subspaces, respectively. Clearly B_1 poorly fits an ideal edge vector, because its projection to the "non-edge" subspace is large. B_1 is rejected by our criterion, whereas it is classified as an edge vector by a conventional threshold decision. Conversely, B_2 is a good fit to the ideal edge element. It is classified as such by the θ criterion but rejected by the conventional decision rule."

Frei and Chen go on to point out that if the non-edge basis vectors described line or point subspaces [see Fig. 5] that the same criterion could be used to find line or point pixels.

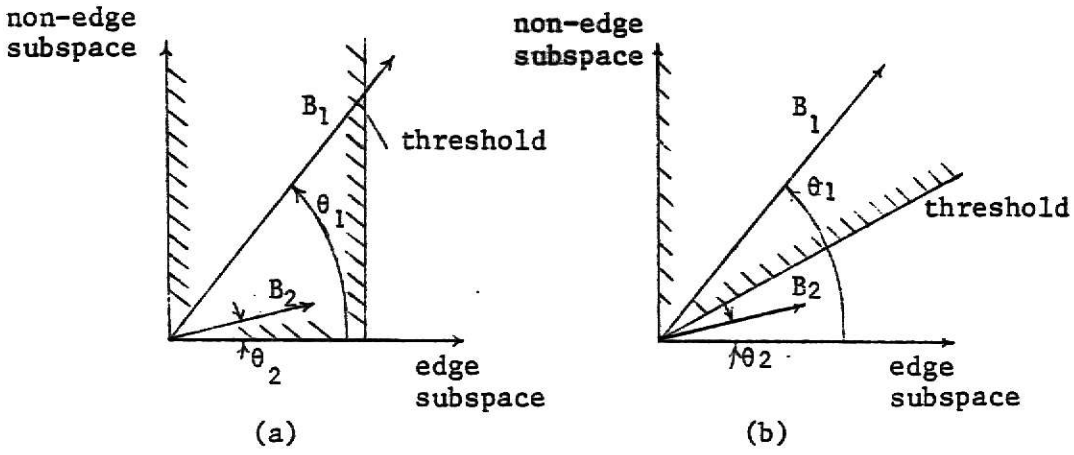


Figure 6. Graphical comparison of (a) conventional method and (b) new boundary classification rule by Frei and Chen (see test) [1].

Another effect as mentioned by Frei and Chen is that equation (5) is invariant to scene illumination. This can be shown by considering the light reaching the camera as a function of illumination I of the scene and the reflectance r_{ij} of the objects in the scene. By substituting r_{ij}*I for pixel grey scale values b_{ij} in equation (5), it can be shown that if r_{ij} is independent of I then I will cancel out of the equation. This means that the new criterion extracts the reflective properties of object boundaries if the small subarea has a constant illumination. The effect is an increased ability to find edges in poorly illuminated areas of an image.

In order to be mathematically precise in equations (3), (4), and (5) it is important to remember that upon implementation the operator vectors must be used in a unit vector form. That is, each operator vector W_i, when used in an equation must satisfy this equation: $(W_i,W_i) = 1$. This is not a hardship as the unit vector of any vector W_i is W_i / $(W_i,W_i)^{1/2}$. This is a point not mentioned by Frei and Chen as they are discussing general concepts but is a necessity when implementing their algorithm.

CHAPTER 6

THE ORTHOGONAL FEATURE BASIS OF FREI AND CHEN

To find a suitable set of orthogonal vectors to be a basis for the n^2-dimensional space, Frei and Chen considered the operator vectors and subspaces for edges and lines used in other algorithms [see Figs. 2, 4]. From the large number of possible orthogonal vectors they chose the nine vectors shown in Fig. 7. These vectors can be grouped to describe three subspaces, the edge, the line and the average subspace. Within each subspace they can be paired on the basis of similarity.

There are two pairs of vectors in the edge subspace. The first pair of vectors W_1 and W_2 are termed the "isotropic gradient" vectors and are taken directly from Fig. 2. The second pair, W_3 and W_4, termed the "ripple" pair, have, to quote the authors, "a distinctive higher order aspect (three zero crossings instead of one)". The origin of these vectors W_3 and W_4 is not clear as they are not similar to any previously defined edge subspace. This may be the reason they are found to contribute little to the edge subspace.

In the line subspace there are also two pairs of vectors. The origin of these vectors is the templates found in Fig. 5(a). The vectors W_5 and W_6, the "line" vectors, which appear to have directional preference are simply $1/3(T_3-T_1)$ and $1/3(T_2-T_4)$ respectively of the vectors in Fig. 5(a). The vectors W_7 and W_8, the "discrete Laplacian"

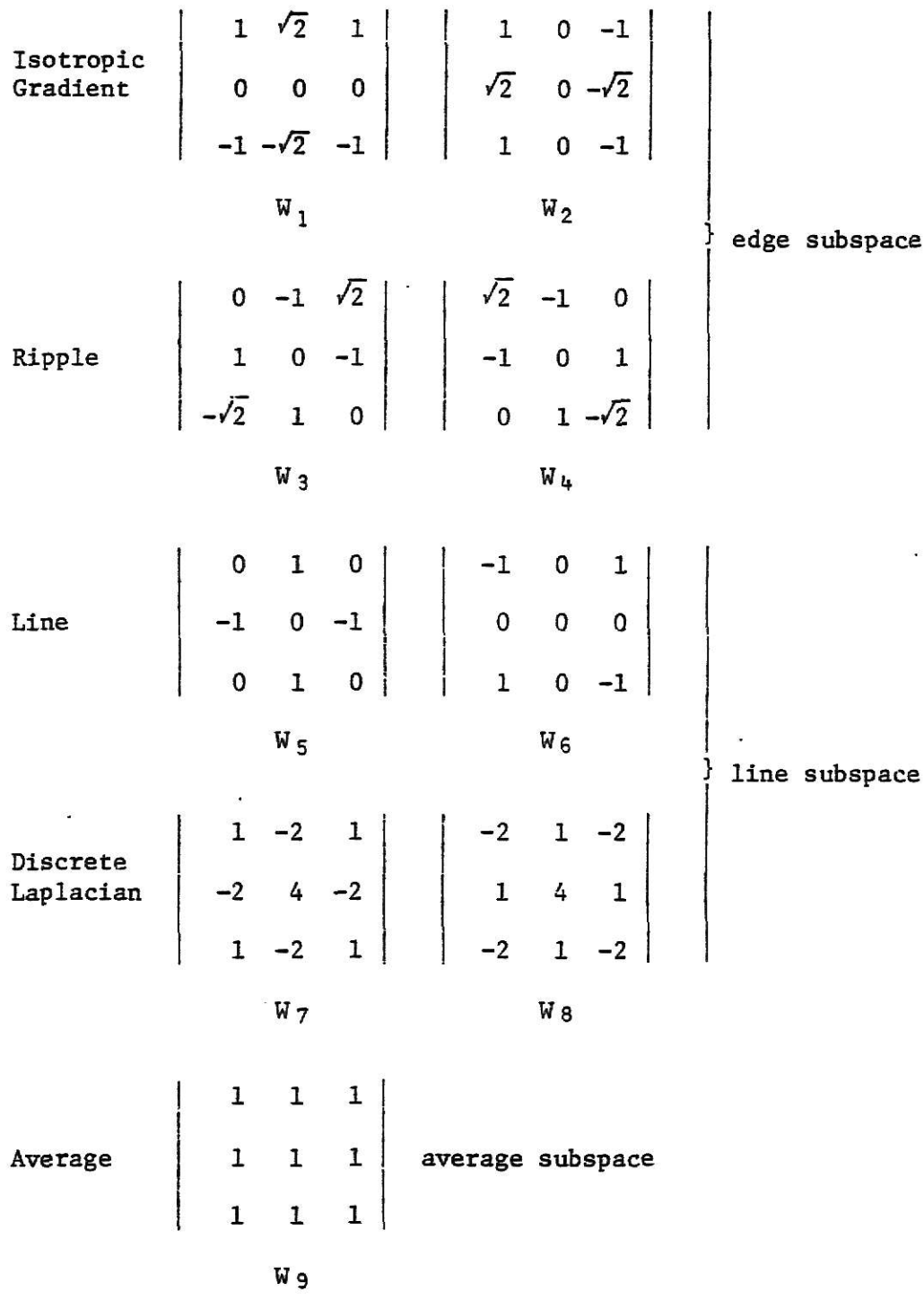


Figure 7. Orthogonal set of "feature" basis vectors chosen by Frei and Chen [1].

vectors, which are without directional preference are (T_2+T_4) and (T_1+T_3). The authors note that the sum (W_7+W_8) is equal to the point vector of Fig. 5(b) and that this pair is a basis for all discrete realizations of the discrete Laplacian [9].

The remaining vector is W_9, the average vector which completes the basis.

CHAPTER 7

SOME RESULTS AND DISCUSSION

Figure 8(a) shows the image of size 512x512 pixels which was used to obtain the results that are presented in what follows.

Tables I, II and III show the variance of the magnitude of projection of the image of Fig. 8(a) onto various subspaces composed of the nine orthogonal basis vectors of Fig. 7. The magnitude of the variance is an indication of the response of the image to the subspace. Table I contains the response of the image to each of the nine orthogonal vectors W_1, W_2, ...W_9 of Fig. 7. Table II contains the variance of the response to the "edge", "ripple", "line", and "discrete Laplacian" pairs of vectors of Fig. 7, and Table III the variance of the response to the "edge" and "line" subspace of Fig. 7.

Three observations, two of which have applications, can be made from these data. By comparing the magnitude of the variances of the projection onto the individual vectors, the pairs of vectors and the subspaces, it is evident that the image has more edge content than line content. A second observation, also based on comparison of the magnitude of the variance, is the minimal contribution, individually and as a pair, of the "ripple" vectors, W_3 and W_4 to the "edge" subspace. Frei and Chen point this out and use it as a reason to exclude these basis vectors from the computation of the "edge" subspace. This reduces computation time with little loss of information.

Magnitude of Projection onto each Vector	Variance of Magnitude of Projection
(B,W_1)	125.65
(B,W_2)	118.95
(B,W_3)	8.56
(B,W_4)	8.26
(B,W_5)	10.66
(B,W_6)	24.61
(B,W_7)	4.43
(B,W_8)	14.34
(B,W_9)	15180.03

Table I. Variance of Magnitude of Projection onto Orthogonal Vectors of Figure 7. B = Image Vector

Magnitude of Projection onto pairs of Vectors	Variance of Magnitude of Projection
$[(B,W_1)^2 + (B,W_2)^2]^{1/2}$	145.94
$[(B,W_3)^2 + (B,W_4)^2]^{1/2}$	5.05
$[(B,W_5)^2 + (B,W_6)^2]^{1/2}$	13.87
$[(B,W_7)^2 + (B,W_8)^2]^{1/2}$	6.71

Table II. Variance of Magnitude of Projection onto pairs of Vectors of Figure 7. B = Image Vector

Magnitude of Projection onto Subspaces	Variance of Magnitude of Projection
Edge Subspace $\left[\sum_{i=1}^{4} (B,W_i)^2\right]^{1/2}$	141.86
Line Subspace $\left[\sum_{i=5}^{8} (B,W_i)^2\right]^{1/2}$	16.87

Table III. Variance of Magnitude of Projection onto "edge" and "line" subspaces of Figure 7. B = Image Vector

A third observation is the large variance of the response to the "average" basis vector W of Fig. 7 in Table I. This variance is also the response to the "average" subspace of Fig. 7. In comparison to the variance magnitudes of the "edge" and "line" subspaces of Table III, it is evident that with most image vectors, the projection onto the "average" subspace is a dominant component. Because of its size it tends to dominate the denominator of equations and compresses the range of the result. The authors note this and provide a new equation to solve this problem. This equation in modified form is

$$\sum_{i=k}^{j} (B,W_i)^2/((B,B)-(B,W_9)^2) \qquad (6)$$

with k and j as appropriate for the desired subspace. The actual equation given by Frei and Chen is

$$\sum_{i=k}^{j} (B,W_i)^2/(B-\bar{B},B-\bar{B}) \qquad (7)$$

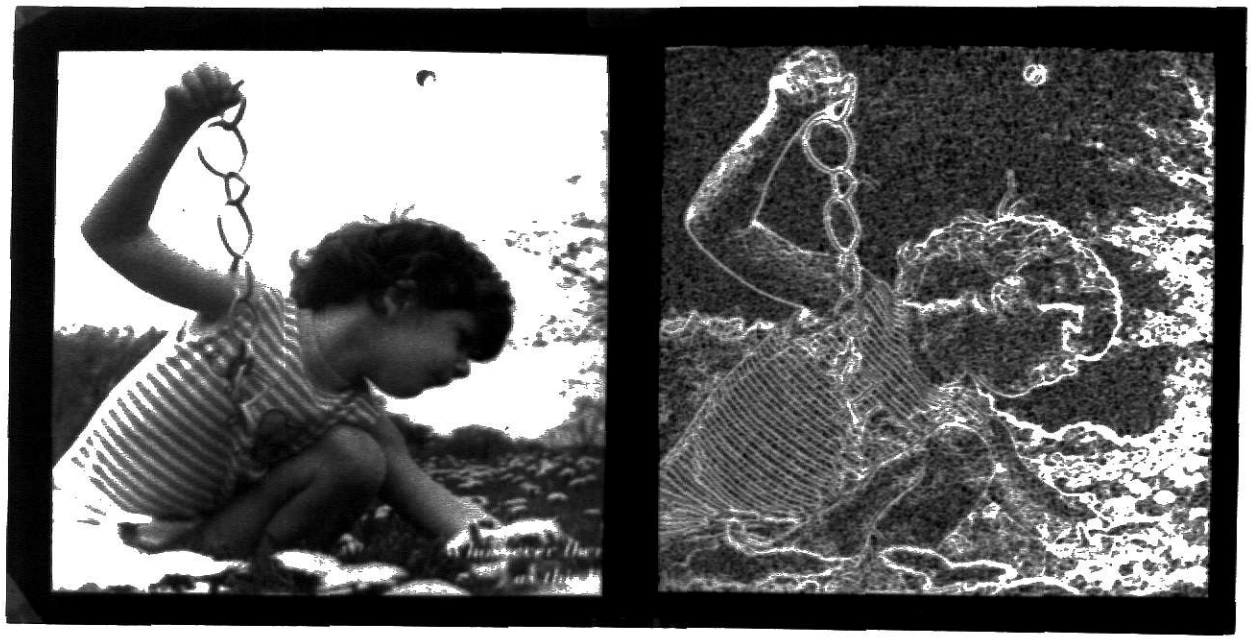
with k and j as appropriate for the desired subspace and $\bar{B}=(B,W_9)$. The denominator of equation (7) is not mathematically precise as $\bar{B}$ is a scalar value and B is a vector. This makes the intention of the denominator $(B-\bar{B},B-\bar{B})$ unclear. The rational for the modified denominator in equation (6) is that since

$$(B,B)=\sum_{i=1}^{4} (B,W_i)^2+\sum_{i=5}^{8} (B,W_i)^2+(B,W_9)^2,$$

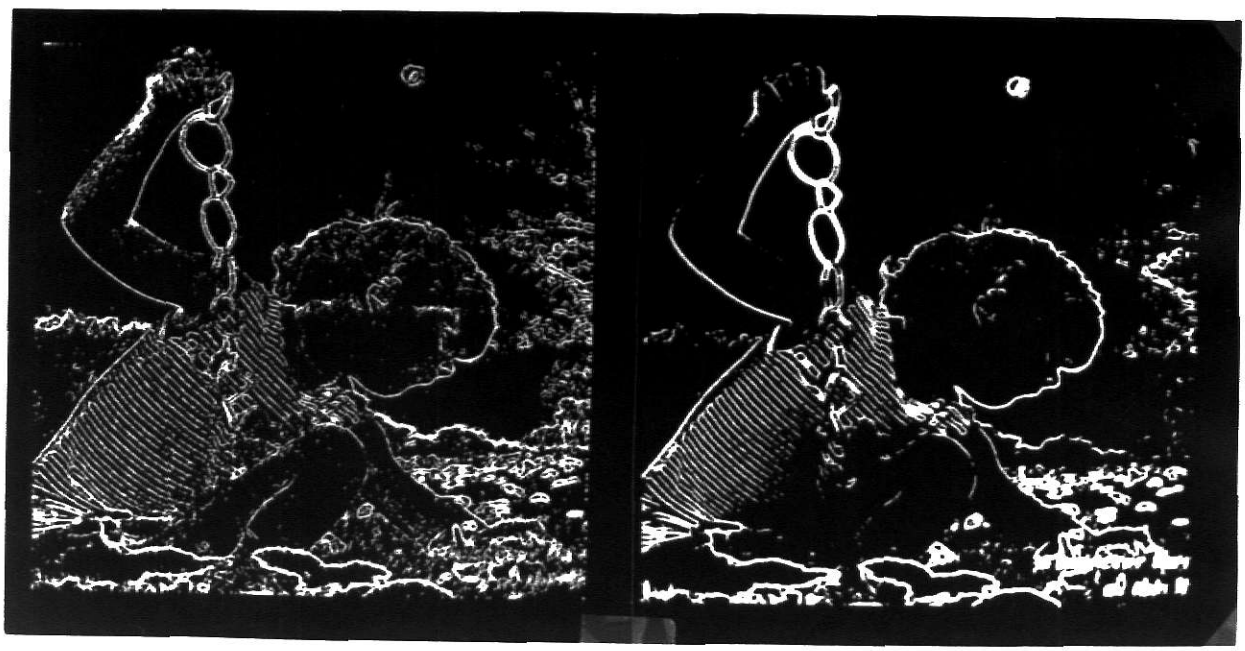
the "average" component of image vector B can be removed from the

denominator by simple substraction of $(B,W_9)^2$. This enhances the range of the result of equation (6) compared to equation (5), as the new denominator is the magnitude of the projection of the image vector onto a new subspace composed of only the "edge" and "line" subspace without consideration for the average subspace. The range is now 0 to 1.

Figure 8(b) shows the result of equation (6) when applied to the image in Fig. 8(a). As Frei and Chen note, their algorithm brings out very fine edge detail even in darker areas of the picture, such as the thigh of the child. This is probably a result of the observation made earlier, that in equation (6) reflectance factors remain while illumination factors cancel. Unfortunately the algorithm, in being sensitive to such edges, is also sensitive to smooth luminance gradients. This can be seen in the sky portion of the picture being cluttered with edge points, on the right forearm of the child and elsewhere. Methods suggested by Frei and Chen of removing erroneous edge points are to reapply their algorithm, but this time find the response of the "edge" image [Fig. 8(b)] to the "line" subspace since the lines of the "edge" image are the edges of the original image. Another method suggested and used by Frei and Chen is to find the response of the "edge" image [Fig. 8(b)] to the "point" vector [Fig. 5(b)]. The point pixels thus detected are then removed from the "edge" image.



(a) (b)



(c) (d)

Figure 8. (a) Original Image,
(b) Results of Frei and Chen algorithm using equ. (6), i=1,j=2,
(c) Image (b) thresholded with 10% of pixels retained,
(d) Thresholded results using Sobel operators of Figure 2 and equ.(1). 10% of the pixels remain.

CHAPTER 8

COMPARISONS

This point detection method was used as the first step in processing the "edge" image of Fig. 8(b) into the image of Fig. 8(c). Approximately five percent of the pixels in the "edge" image, that responded the strongest to the "point" template of Fig. 5(b), had their grey scale value zeroed. Then the resulting image was thresholded so that approximately ten percent of the pixels remained. The result is Fig. 8(c).

Figure 8(a) is an image for comparison to Fig. 8(c). It is the result of the original image in Fig. 8(a) being projected, using equation (1), onto the subspace formed by the two Sobel operators found in Fig. 2. The resulting image was thresholded so that as with Fig. 8(c) approximately ten percent of the pixels remained.

Figure 9(a) is another image, this time of size 256x256 pixels, to which the algorithm of Frei and Chen was applied. It is an infrared image of a man standing beside an elevated box with a wire fence behind and to his right. The unprocessed result of the algorithm of Frei and Chen can be seen in Fig. 9(b). The image in Fig. 9(c) is the image of Fig. 9(b) processed the same way as the image of Fig. 8(c). And for comparison Fig. 9(a) shows the original image processed in the same way as Fig. 8(d).

A comparison of the images in Fig. 8(c) and Fig. 8(d), Fig. 9(c)

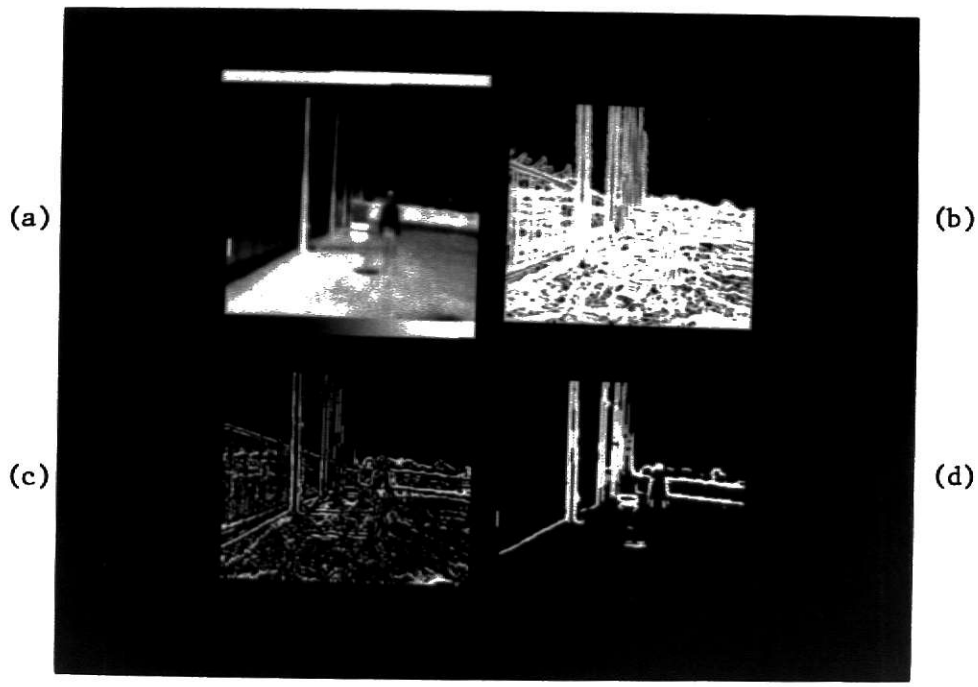


Figure 9. (a) Original Infra-Red Image,
(b) Image (a) processed with Frei and Chen algorithm using equ. (6) with i = 1, j = 2,
(c) Image (b) thresholded to retain 10% of pixels,
(d) Image (a) processed with Sobel operators of Figure 2 and equ. (1), then thresholded to retain 10% of pixels.

and Fig. 9(a) reveals that for approximately the same number of pixels, more detail is visible using the algorithm of Frei and Chen. Advantages listed by Frei and Chen of their algorithm as "clearly revealed" by the results are:

1) much more subtle edges are detected,
2) "strong" edges are detected as thinner lines (minimizing the need for "thinning" operations),
3) edges in dark areas of the image are more likely to be detected.

Disadvantages are an increased sensitivity to noise and additional computational time.

CHAPTER 9

MORE RESULTS AND DISCUSSION

For a more thorough understanding of the algorithm of Frei and Chen, a computer program was written [See Appendix B] which produces an ideal edge, at an arbitrary angle, in an image subarea. More images are generated as this edge is shifted away from the center of the subarea. The result is a sequence of images of an ideal edge shifting across the subarea. This sequence of images was then enhanced using the algorithm of Frei and Chen and the response studied. The idea for this program came from a paper by Abdou and Pratt [7].

The response of the algorithm of Frei and Chen to these generated images shows several properties of the algorithm. One property is that the response is invariant to the average value of the pixels in the generated images. This was tested by adding a constant to the generated images and resulted in no change in the response. This was expected and is also a property of the algorithm of Sobel [4].

Another property is that the response is not a function of the magnitude of the edge. As defined previously, the magnitude is the difference in the grey scale values of the pixels on either side of the edge. This property was tested by multiplying the generated images by a constant. This multiplication increased the difference in grey scale values but resulted in no change in the response. The response of the algorithm of Sobel [4], which uses equation (1), did

change and was proportional to the magnitude.

This property of invariance to magnitude partially explains several characteristics of the algorithm of Frei and Chen. It explains why fine detail edges and edges in dark areas of the picture with magnitudes of, for example, 2 are enhanced equally with strong edges in lighter portions of the image with magnitudes of, for example, 150. It also explains why smooth luminance gradients which can be identical to edges with small magnitudes, are enhanced and why pixel values in a subarea need only be changed slightly by noise for the subarea to be enhanced as an edge. Of course, this property is based on the mathematics of equation (6). Evidently the effects of magnitude are constants which can be separated from the numerator and denominator of equation (6) and cancelled.

The response of the algorithm of Frei and Chen to images of shifted edges, reveals that the response is a function of the distance from the edge to the center of the subarea and of the edge angle. The response is shown in Fig. 10 to the subarea images of a vertical edge rotated 0 degrees, 23 degrees and 45 degrees respectively, as the edge is shifted away from the center of the subarea. The horizontal axis of Fig. 10 indicates in fractions of one pixel width the distance the edge has been shifted. These responses lead to the conclusion that the criterion for edge enhancement is based on the position of the zero crossing point between pixel values of the subarea, once the average value of the pixels has been removed. The position of zero crossing can be determined by comparing the power of the cosine and sine waveforms which sum to form the image subarea.

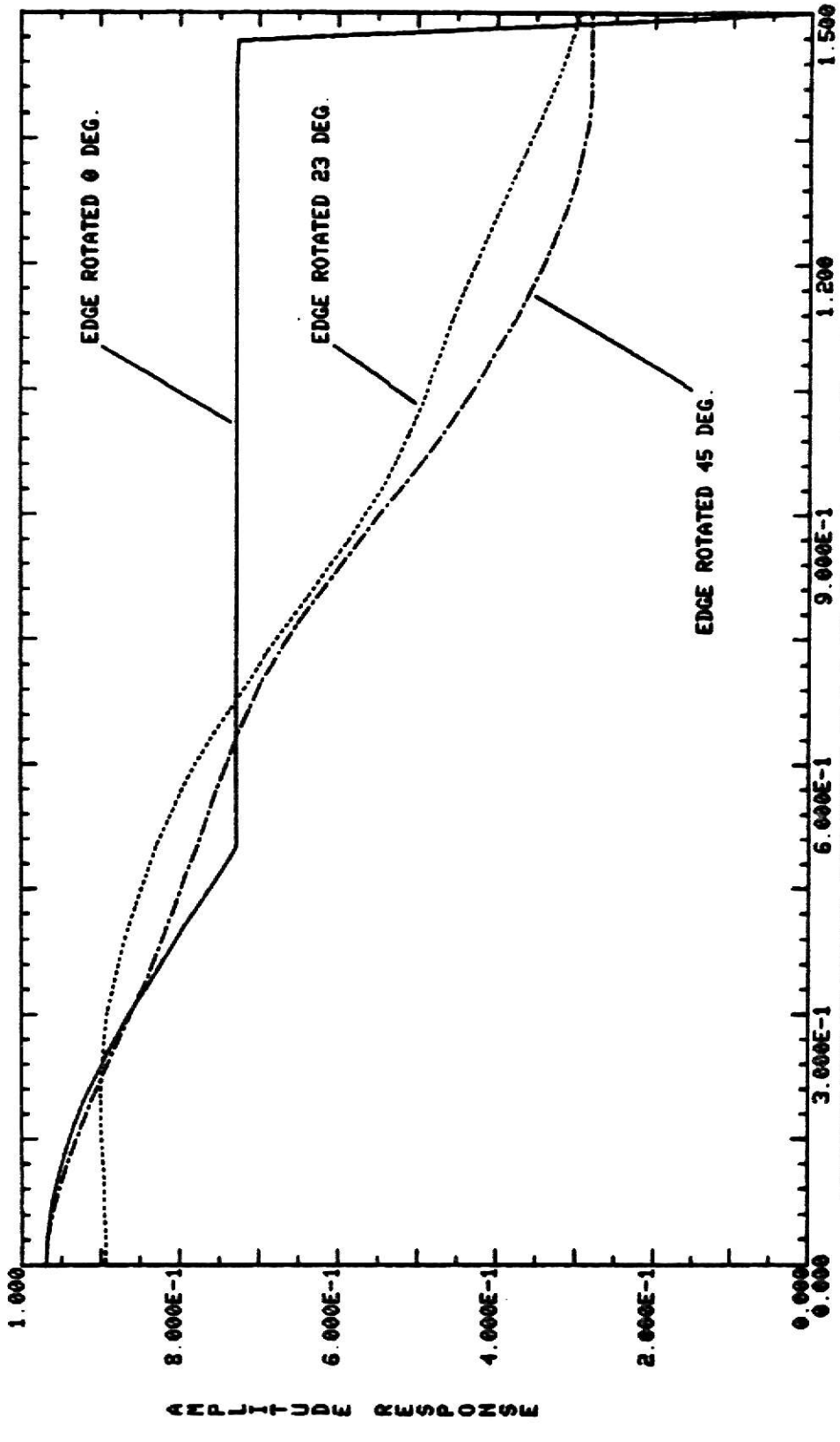


Fig. 10. Amplitude response of algorithm of Frei and Chen to three rotated ideal edges as they are shifted from the center of a 3x3 pixel subarea.

CHAPTER 10

COSINE AND SINE COMPONENTS OF A 3x3 SUBAREA

To understand the cosine and sine components of an image subarea, it is necessary to first consider a narrow strip of an image in the horizontal, vertical or diagonal direction. If this strip is considered as a continuous signal $x(\xi)$ where ξ is distance along the strip and $x(\xi)$ is considered periodic, then $x(\xi)$ can be expanded into the sum of an infinite number of harmonically related cosine and sine terms; i.e.,

$$x(\xi) = a_0/2 + \sum_{m=1}^{\infty} (a_m \cos m\omega_1\xi + b_m \sin m\omega_1\xi) \qquad (8)$$

where

$$a_m = (2/NS_i) \int_{-NS_i/2}^{NS_i/2} x(\xi) \cos m\omega_1\xi \, d\xi \qquad (9)$$

and

$$b_m = (2/NS_i) \int_{-NS_i/2}^{NS_i/2} x(\xi) \sin m\omega_1\xi \, d\xi \qquad (10)$$

The terms in this expansion are defined as follows:

NS_i = period of waveform (N = # of pixels, S_i = pixel width),

f_1 = fundamental cyclic frequency = $1/NS_i$,

ω_1 = fundamental radian frequency = $2\pi f_1$,

m = integer defining order of harmonic.

When this strip $x(\xi)$ is digitized, aliasing and folding over occurs for frequencies above $1/2S_i$. This means that if the digitized signal was used to regenerate a continuous signal the resulting signal $x'(\xi)$ could now be expanded into the sum of a finite number of harmonically related cosine and sine terms. How close $x'(\xi)$ depends on the original frequency bandwidth of $x(\xi)$.

If $x(\xi)$ is considered one row of an image and is copied into every other row of the image then the resulting image $I(\xi)$ could be expanded into an infinite sum of cosine and sine images which vary only with row distance ξ. If this continuous image is digitized and the continuous signal regenerated from the digital values, the resulting image $I'(\xi)$ could be expanded into a finite sum of harmonically related cosine and sine images. The same can be said if $x(\xi)$ is considered a column or a diagonal strip of an image.

In the case of rows or columns, the fundamental harmonic frequency of the cosine and sine images will be $1/NS_i$, where N is the number of pixels in the row or column and S_i is the pixel width or sampling interval. This sampling interval S_i also limits the maximum harmonic frequency of the finite sum of cosine and sine images to $1/2S_i$.

In the case where $x(\xi)$ is a diagonal strip of an image, the sampling interval is not the same as along a row or column. If the sampling interval along a row is S_i then the sampling interval along a diagonal is $S_i' = (\sqrt{2}/2)S_i$. The reason is that samples of the image taken anywhere along a perpendicular line to the diagonal are true samples of the image along the diagonal. They are true samples because the diagonal strip $x(\xi)$ was copied to adjacent diagonal strips.

If an NxN pixel image is assumed, then the number of samples in the diagonal direction is 2N and the fundamental harmonic frequency is $1/2NS_i'$. The maximum harmonic frequency is $1/2S_i'$.

Figure 12 shows the resulting pixel values of a 3x3 subarea of each of the harmonic cosine and sine images along a row. They are digital samples of each of the harmonic images, referenced so that the cosine waveform crosses zero at the center of the subarea.

Because they are orthogonal, the three masks M_1, M_2, and M_3 of Figure 12 can be called a basis set for any possible subarea which varies only along its row distance. These three masks, developed from knowledge of the harmonic frequency components of the total image, imply that any possible image subarea which varies only along its rows can be considered as a combination of an average constant, a cosine waveform of harmonic frequency less than or equal to $1/NS_i$ and which crosses zero at the center of the subarea, and a sine waveform of equal frequency. The power relationship between the cosine and sine component is thus an indication of the position of the zero crossing point in an image subarea. These three masks, if rotated ninety degrees, generate the same results for an image which varies only along its columns.

Figure 13 shows the resulting pixel values of a 3x3 pixel subarea of each of the harmonic cosine and sine images possible that vary according to diagonal distance. The cosine and sine masks are orthogonal to each other but masks within each category are not. Part of the reason for this is that an inner product, in this case, is a discrete integration of the product of two sinusoids of different frequency. The integration result is not zero because the integration

Frequency ($f=1/NS_i$)	Cosine	Sine
0	$\begin{vmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{vmatrix}$	$\begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$
		M_1: 0 zero crossing
$0 < nf < 1/2S_i$	$\begin{vmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{vmatrix}$	$\begin{vmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{vmatrix}$
	M_2: 1 zero crossing	
$1/2S_i$	$\begin{vmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{vmatrix}$	$\begin{vmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{vmatrix}$
		M_3: 2 zero crossings

Figure 12. Digitized cosine and sine images in row direction.

Frequency ($f=1/2NS_i'$)	Cosine	Sine
0	$\begin{vmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{vmatrix}$	$\begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$
		Z_1: 0 zero crossing
$0 < nf < 1/5S_i'$	$\begin{vmatrix} 0.00 & 0.95 & 0.59 \\ -0.95 & 0.00 & 0.95 \\ -0.59 & -0.95 & 0.00 \end{vmatrix}$	$\begin{vmatrix} 0.72 & 0.01 & -1.10 \\ 0.01 & 0.72 & 0.01 \\ -1.10 & 0.01 & 0.72 \end{vmatrix}$
	Z_2: 1 0 crossing	Z_3: 2 0 crossings
$1/4S_i'$	$\begin{vmatrix} 0 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & 0 \end{vmatrix}$	$\begin{vmatrix} 8 & -1 & -10 \\ -1 & 8 & -1 \\ -10 & -1 & 8 \end{vmatrix}$
	Z_4: 2 0 crossings	Z_5: 2 0 crossings
$1/3S_i'$	$\begin{vmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{vmatrix}$	$\begin{vmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{vmatrix}$
	Z_6: 3 0 crossings	Z_7: 2 0 crossings
$1/2S_i'$	$\begin{vmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{vmatrix}$	$\begin{vmatrix} 4 & -5 & 4 \\ -5 & 4 & -5 \\ 4 & -5 & 4 \end{vmatrix}$
		Z_8: 4 0 crossings

Figure 13. Digitized cosine and sine images in one diagonal direction.

distance is not a complete number of cycles for each frequency. Another fact to note is that images that vary diagonally also vary in the row and column direction. This means that some of the diagonal subareas are not orthogonal to some of the row subareas. However because the cosine and sine subareas of Figure 13 are orthogonal there is again an indication of the possibility of separating a diagonal image subarea into three general components, an average value, a cosine waveform which crosses zero in the middle of the subarea and a sine waveform of equal frequency. And as before the power relationship between the cosine and sine components gives an indication of how close is the zero crossing to the center of the subarea.

If nine orthogonal subareas are selected from those in Figures 12 and 13, the most obvious ones to choose first are M_1, M_2, M_2 rotated 90°, M_3 and M_3 rotated 90° of Figure 12. The only subareas of Figure 13 which are orthogonal are Z_6, Z_6 rotated 90°, Z_7 and Z_7 rotated 90°. These nine orthogonal basis vectors are shown in Figure 14 as vectors D_1, D_2,.., D_9.

The relationship between the "cosine" vectors D_1,.., D_4 of Figure 14 and the edge subspace vectors W_1,.., W_4 of Figure 7 is fairly clear. Both sets look like cosine waveforms that cross zero at the center of the subarea. W_1 and W_2 look like cosine waveforms in the column and row direction. The sum $(W_1 + W_2)$ looks like a cosine waveform in the diagonal direction with one zero crossing and W_3 and W_4 look like cosine waveforms in the diagonal directions with three zero crossings.

The relationship between the "sine" vectors D_5,.., D_8 of Figure 14 and the line subspace vectors W_5,.., W_8 of Figure 7 is more tenuous

$$D_1 = \begin{vmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{vmatrix} \qquad D_2 = \begin{vmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{vmatrix}$$

$$D_3 = \begin{vmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{vmatrix} \qquad D_4 = \begin{vmatrix} -1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & 1 \end{vmatrix}$$

} "cosine" vectors

$$D_5 = \begin{vmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{vmatrix} \qquad D_6 = \begin{vmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{vmatrix}$$

$$D_7 = \begin{vmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{vmatrix} \qquad D_8 = \begin{vmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{vmatrix}$$

} "sine" vectors

$$D_9 = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

Figure 14. Nine orthogonal subarea vectors taken from Figures 12 and 13.

but also direct. $W_5 = (1/3)(D_5-D_6)$, $W_6 = (1/3)(D_7-D_8)$, $W_7 = (D_7+D_8)$, and $W_8 = (D_5+D_6)$. Both sets are similar to sine waves with peaks at at the center of the subarea. W_5 and W_8 look similar to sine waveforms in the row and column direction and W_6 and W_7 look similar to sine waves in the diagonal direction with W_7 being of higher frequency as there are four zero crossings.

CHAPTER 11

CONCLUSIONS

With the relationships between "cosine" and "edge" vectors and between "sine" and "line" vectors of chapter 10 in mind, an explanation for the edge detection criterion of Frei and Chen can be given. Due to the limitations of only three pixel values in the row and column direction and five sample values in the diagonal direction, all information in 3x3 image subareas can be expanded as the finite sum of cosine and sine waveforms. Cosine waveforms are defined as having a zero crossing at the center of the subarea and sine waveforms as having a peak. There is one cosine and one sine waveform for rows, for columns and for each of the diagonal directions.

As the edge subspace of Figure 7, chosen by Frei and Chen, consists of all cosine waveforms, the implication is that an ideal edge subarea can be expanded with only cosine terms. The criterion for edge detection found in equation (6) can now be explained as a ratio of the power in the cosine waveforms of a subimage versus the total power found in the cosine and sine waveforms. This ratio is an indication of how close the zero crossing is to the center of the subarea.

As the line subspace of Figure 7 consists of all sine waveforms the implication is that an ideal line subarea can be expanded as a finite sum of sine waveforms. The criterion for line detection can

then be the ratio of the power in the sine waveforms versus the total power in the sine and cosine waveforms.

The advantages of this power ratio for edge detection are "thinner" edges due to greater sensitivity to actual edge position in a subarea and true edges with small magnitudes, such as those in dark areas of an image, being detected.

Some disadvantages, because of sensitivity to actual edge position and not to magnitude, is sensitivity to smooth luminance gradients and to noise. A possible method to lessen this problem in addition to those methods listed by Frei and Chen is to use two thresholds. One threshold is used for the magnitude of an edge. This threshold is set to eliminate the small magnitudes of smooth luminance gradients and noise with minimum loss of true edge information. The other threshold is used for the power ratio between cosine waveforms and total sinusoid power. This ratio is set to eliminate edges which are not close to the center of the image subarea.

REFERENCES

[1] Frei, W. and Chen, C., "Fast Boundary Detection: A Generalization and a New Algorithm", I.E.E.E. Transactions on Computers, Vol C-26, No. 10, October 1977.

[2] Roberts, L.G., "Machine Perception of Three-Dimensional Solids", Optical and Electro-Optical Information Processing, J.T. Tippett et. al. eds., Cambridge, Ma, MIT Press 1965.

[3] Kirsch, R., "Computer Determination of the Constituent Structures of Biological Images", Computer Biomedical Research, vol 4, pp 315-328, 1971.

[4] Duda, R.O. and Hart, P.E., Pattern Classification and Scene Analysis, New York: Wiley, 1971.

[5] Prewitt, J.M.S., "Object Enhancement and Extraction", Picture Processing and Psychopictorics, Lipkin, B.S. and Rosenfeld, A. eds., New York: Academic Press,1970, pp.75-149.

[6] Robinson, G.S., "Detection and Coding of Edges using Directional Masks", Univ. of Southern California, Los Angeles, CA, USC-IPL Rep 660, pp. 40-57, Mar. 1976.

[7] Abdou, I.E. and Pratt, W.K., "Quantitative Design and Evaluation of Enhancement/Thresholding Edge Detectors", Proc. of the I.E.E.E., vol. 67, No. 5, May 1979.

[8] Agnew, J. and Knopp, R.C., Linear Algebra with Applications, Brooks/Cole Publishing Co., Monterey CA, 1978.

[9] Frei, W. "A New Class of Edge and Feature Detection Operators", Univ. of Southern Cal., Los Angeles, Ca, USC-IPL Rep. 660, pp. 22-40, Mar. 1976.

APPENDIX A

This appendix contains the Fortran listings of five programs used on the Data General VAX computer of the Engineering College of Kansas State University. These programs let the user generate vector files, find the responce of images to these vectors and display the results.

```
C       PROGRAM CRMASK.FOR
C
C       THIS PROGRAM LETS THE USER CREATE AND MODIFY
C       9 ELEMENT DATA FILES USED AS MASK VECTORS IN
C       PROGRAM FRECHEN.FOR.
C
        REAL*4 AR(9)
        CHARACTER*10 CHO
        LOGICAL ANS
        IO1=1
        IO2=2
        NPOINT = 9
C
        CALL REPLY(' MODIFY OLD FILE (Y/N)? ',ANS)
        IF (ANS)
     &     THEN
              CALL SGOPEN(IO1,'READ',' OLD FILE? ','NONAME','REAL',NPT)
              CALL SGTRAN(IO1,'READ','REAL',AR,NPOINT)
        END IF
C
        CALL SGOPEN(IO2,'WRITE',' NEW FILE? ','NONAME','REAL',NPOINT)
C
 10     CONTINUE
 100    FORMAT(A)
        TYPE 100,'$ENTER,PRINT,MODIFY OR QUIT (E,P,M,Q)? '
        ACCEPT 100,CHO
C
        IF (CHO(1:1).EQ.'P') THEN
 101       FORMAT ('$ ARRAY(',I1,') = ',F15.9)
           DO I = 1,9
              TYPE 101,I,AR(I)
           END DO
           GO TO 10
        END IF
C
        IF (CHO(1:1).EQ.'M') THEN
 15        CONTINUE
           TYPE 100,'$WHICH ELEMENT TO CHANGE (1-9)? '
           ACCEPT *,NELE
           IF (NELE.GT.9.OR.NELE.LT.1) GO TO 15
 102       FORMAT ('$ ARRAY(',I1,') = ')
           TYPE 102,NELE
           ACCEPT *,AR(NELE)
           GO TO 10
        END IF
C
        IF (CHO(1:1).EQ.'E') THEN
           DO I=1,9
              TYPE 102,I
              ACCEPT *,AR(I)
           END DO
```

```
            GO TO 10
         END IF
C
         CALL SGTRAN(IO2,'WRITE','REAL',AR,NPOINT)
C
         END
```

```
C       PROGRAM INVAR.FOR
C
C       THIS PROGRAM FINDS THE MAXIMUM VALUE, THE MINIMUM
C       VALUE, THE MEAN AND VARIANCE OF THE RESPONSE OF
C       A OPERATOR TO AN IMAGE.
        REAL*8 SUMSQ,SUM
        REAL RMIN,RMAX,POINTS,RMEAN,DX
        INTEGER*2 IMA1(512,512)
        INTEGER NN,MM,NUM,NPOINT
        REAL MSK(9,9),LEN(9),R1,RS,RT
        LOGICAL ANS

        MSKIOI = 1
        IMIOI = 2
C
C       INPUT SUBSPACE VECTORS
C
        CALL SGOPEN(MSKIOI,'READ',' VECTOR FILE ? ','NONAME','REAL',NPOINT)
        CALL SGTRAN(MSKIOI,'READ','REAL',MSK,NPOINT)
        NUM = NPOINT/9
C
C       INPUT IMAGE FILENAMES
C
        CALL IMOPEN(IMIOI,'READ','INPUT IMAGE FILE? ','NONAME',NN,MM)
C
        CALL IMTRAN(IMIOI,'READ','INTEGER*2',IMA1,512,NN,MM)
C
C       INPUT DIFFERENT PICTURE SIZES IF DESIRED
C
        IRST = 1
        ICST = 1
        IREN = MM
        ICEN = NN
        TYPE *
        TYPE *,' NUMBER OF ROWS = ',MM
        TYPE *,' NUMBER OF COLUMNS = ',NN
        CALL REPLY(' IS THE REAL PICTURE SMALLER (Y/N)? ',ANS)
        IF (ANS) THEN
          CALL INPUT(' STARTING ROW? ',1,MM,IRST)
          CALL INPUT(' ENDING ROW? ',IRST,MM,IREN)
          CALL INPUT(' STARTING COLUMN? ',1,NN,ICST)
          CALL INPUT(' ENDING COLUMN? ',ICST,NN,ICEN)
        END IF

C
C       FIND LENGTH OF BASIS VECTORS
C
        DO M1=1,NUM
           LEN(M1) = 0.0
           DO M2=1,9
              LEN(M1) = LEN(M1) + MSK(M2,M1)*MSK(M2,M1)
```

```
            END DO
            LEN(M1) = LEN(M1)**0.5
          END DO
C
C         FIRST FIND MAGNITUDE
C
          SUMSQ = 0.0
          SUM = 0.0
          RMAX = -10000.0
          RMIN = 10000
          RT = 0.0
          DO 200 LX = IRST,IREN-2
           DO 100 LY = ICST,ICEN-2
             LX1 = LX + 1
             LX2 = LX + 2
             LY1 = LY + 1
             LY2 = LY + 2
C
             RS = 0.0
C
            DO M1=1,NUM
             R1 = MSK(1,M1)*IMA1(LY,LX) + MSK(2,M1)*IMA1(LY1,LX) +
     &           MSK(3,M1)*IMA1(LY2,LX) + MSK(4,M1)*IMA1(LY,LX1) +
     &           MSK(5,M1)*IMA1(LY1,LX1) + MSK(6,M1)*IMA1(LY2,LX1) +
     &           MSK(7,M1)*IMA1(LY,LX2) + MSK(8,M1)*IMA1(LY1,LX2) +
     &           MSK(9,M1)*IMA1(LY2,LX2)
C
             R1 = R1 / LEN(M1)
             RS = RS + R1 * R1
            END DO
C
            RS = RS**0.5
            IF(NUM.EQ.1) RS = R1
            SUM = SUM + RS
            SUMSQ = SUMSQ + RS*RS
            RMAX = AMAX1(RMAX,RS)
            RMIN = AMIN1(RMIN,RS)
 100       CONTINUE
 200      CONTINUE
C
          POINTS = (IREN-IRST+1)*(ICEN-ICST+1)
          RMEAN = SUM / POINTS
          RVAR = SUMSQ / POINTS - SUM*SUM / (POINTS*POINTS)
C
          TYPE *,' MAXIMUM = ',RMAX
          TYPE *,' MINIMUM = ',RMIN
          TYPE *,' MEAN = ',RMEAN
          TYPE *,' VARIANCE = ',RVAR
C
          END
```

```
C************************************************
C************************************************
C
C       INPUT BOUNDED INTEGER FROM TERMINAL
C
        SUBROUTINE INPUT (PROMPT, FIRST, LAST, VALUE)
C
        IMPLICIT NONE
        CHARACTER*(*)  PROMPT
        INTEGER  FIRST, LAST, VALUE
        LOGICAL  VALID
C
1       FORMAT (A)
C
        VALID = .FALSE.
        DO  WHILE (.NOT.VALID)
            TYPE 1, '$'//PROMPT
            READ *, VALUE
            IF (VALUE.LT.FIRST)  THEN
                TYPE *
                TYPE *, 'response can not be smaller than -> ', FIRST
                TYPE *
            ELSE    IF (VALUE.GT.LAST)  THEN
                TYPE *
                TYPE *, 'response can not be larger than -> ', LAST
                TYPE *
            ELSE
                VALID = .TRUE.
            END IF
        END DO
        RETURN
        END
```

```
C       PROGRAM IMHIST.FOR
C
C       THIS PROGRAM INPUTS AN IMAGE FILE AND FINDS
C       THE HISTOGRAM.  ON REQUEST IT WILL RETURN
C       TO THE USER THE PERCENT OF TOTAL POINTS THAT
C       ARE ABOVE THE INPUTED CUTOFF VALUE.
C       WHEN A DESIRED CUTOFF VALUE IS REACHED THE
C       IMAGE IS BINARIZED AROUND THE CUTOFF VALUE.
C       BINARIZED MEANS ALL VALUES BELOW OR EQUAL TO
C       CUTOFF ARE SET TO ZERO AND THOSE ABOVE ARE SET
C       TO 255.
C
        LOGICAL ANS
        REAL HIST(0:255),SUM,TOT,PER
        INTEGER*2 IMA1(512,512),IMA2(512,512)
        INTEGER CVAL,HDISP,VDISP,NN,MM
C
        IMIOI = 1
        IMIOO = 2
C
        CALL IMOPEN(IMIOI,'READ',' INPUT FILE? ','NONAME',NN,MM)
        CALL IMOPEN(IMIOO,'WRITE',' OUTPUT FILE? ','NONAME',NN,MM)
C
C       FIND HISTOGRAM
C
        CALL IMTRAN(IMIOI,'READ','INTEGER*2',IMA1,512,NN,MM)
C
        DO LX=0,255
          HIST(LX) = 0.0
        END DO
C
C
C       INPUT DIFFERENT PICTURE SIZES IF DESIRED
C
        IRST = 1
        ICST = 1
        IREN = MM
        ICEN = NN
        TYPE *
        TYPE *,' NUMBER OF ROWS = ',MM
        TYPE *,' NUMBER OF COLUMNS = ',NN
        CALL REPLY(' IS THE REAL PICTURE SMALLER (Y/N)? ',ANS)
        IF (ANS) THEN
          CALL INPUT(' STARTING ROW? ',1,MM,IRST)
          CALL INPUT(' ENDING ROW? ',IRST,MM,IREN)
          CALL INPUT(' STARTING COLUMN? ',1,NN,ICST)
          CALL INPUT(' ENDING COLUMN? ',ICST,NN,ICEN)
        END IF

        DO LX = IRST,IREN
          DO LY = ICST,ICEN
```

```
            I = IMA1(LY,LX)
            HIST(I) = HIST(I) + 1
          END DO
        END DO
C
        TOT = (IREN-IRST+1)*(ICEN-ICST+1)
 10     CONTINUE
        CALL INPUT(' CUTOFF VALUE (0->254)? ',0,254,CVAL)
C
        SUM = 0.0
C
        DO LX = CVAL+1,255
           SUM = SUM + HIST(LX)
        END DO
C
        PER = (SUM/TOT) * 100.0
C
 99     FORMAT(' CUTOFF =',I4,'  PERCENT = ',F9.2)
        TYPE 99,CVAL,PER
C
        CALL REPLY(' NEW CUTOFF PERCENT (Y/N)? ',ANS)
        IF (ANS) GO TO 10
C
        DO LX = IRST,IREN
          DO LY = ICST,ICEN
            IF(IMA1(LY,LX).GT.CVAL)
     &        THEN
              IMA2(LY,LX) = 255
            ELSE
              IMA2(LY,LX) = 0.0
            END IF
          END DO
        END DO
C
        CALL REPLY(' CLEAR SCREEN (Y/N)? ',ANS)
        IF (ANS)
     &    THEN
            CALL IMINIT('ERASE')
          ELSE
            CALL IMINIT('NOERASE')
        END IF
C
        CALL INPUT(' HORIZONTAL DISPLACEMENT (0-256)? ',0,256,HDISP)
        CALL INPUT(' VERTICAL DISPLACEMENT (0-256)? ',0,256,VDISP)
C
        CALL IMDISP('WRITE','INTEGER*2',IMA2,512,NN,MM,HDISP,VDISP,'WHITE')
C
        CALL REPLY(' TRY NEW CUTOFF (Y/N)? ',ANS)
        IF (ANS) GO TO 10
C
        CALL IMTRAN(IMIOO,'WRITE','INTEGER*2',IMA2,512,NN,MM)
        END
```

```
C*************************************************
C*************************************************
C
C       INPUT BOUNDED INTEGER FROM TERMINAL
C
        SUBROUTINE INPUT (PROMPT, FIRST, LAST, VALUE)
C
        IMPLICIT NONE
        CHARACTER*(*)  PROMPT
        INTEGER  FIRST, LAST, VALUE
        LOGICAL  VALID
C
1       FORMAT (A)
C
        VALID = .FALSE.
        DO  WHILE (.NOT.VALID)
            TYPE 1, '$'//PROMPT
            READ *, VALUE
            IF (VALUE.LT.FIRST)  THEN
                TYPE *
                TYPE *, 'response can not be smaller than -> ', FIRST
                TYPE *
            ELSE    IF (VALUE.GT.LAST)  THEN
                TYPE *
                TYPE *, 'response can not be larger than -> ', LAST
                TYPE *
            ELSE
                VALID = .TRUE.
            END IF
        END DO
        RETURN
        END
```

```
C       PROGRAM IMCOM.FOR
C
C       THIS PROGRAM INPUTS TWO IMAGE FILES.  IT COMPARES THE
C       VALUES IN THE SECOND IMAGE TO A CUTOFF VALUE AND IF IT\
C       IS GREATER THAN CUTOFF IT ZEROS THE VALUE IN THE
C       FIRST IMAGE FILE.  IT DISPLAYS THE RESULTING IMAGE
C       FILE AND THEN IF DESIRED WILL OUTPUT THE RESULT.
C       DIFFERENT CUTOFF VALUES CAN BE USED AND THE RESULTS
C       DISPLAYED BUT BECAUSE OF THE WAY THE PROGRAM WORKS
C       START WITH LARGEST CUTOFF VALUE AND WORK YOUR WAY
C       WAY DOWN.
C
        INTEGER*2 IMA1(512,512),IMA2(512,512)
        INTEGER CVAL,NN,MM
        LOGICAL ANS
C
        IMIOI1 = 1
        IMIOI2 = 2
        IMIOO = 3
C
        CALL IMOPEN(IMIOI1,'READ',' FILE #1? ','NONAME',NN,MM)
        CALL IMOPEN(IMIOI2,'READ',' FILE #2? ','NONAME',NN,MM)
C
C       INPUT DIFFERENT PICTURE SIZES IF DESIRED
C
        IRST = 1
        ICST = 1
        IREN = MM
        ICEN = NN
        TYPE *
        TYPE *,' NUMBER OF ROWS = ',MM
        TYPE *,' NUMBER OF COLUMNS = ',NN
        CALL REPLY(' IS THE REAL PICTURE SMALLER (Y/N)? ',ANS)
        IF (ANS) THEN
          CALL INPUT(' STARTING ROW? ',1,MM,IRST)
          CALL INPUT(' ENDING ROW? ',IRST,MM,IREN)
          CALL INPUT(' STARTING COLUMN? ',1,NN,ICST)
          CALL INPUT(' ENDING COLUMN? ',ICST,NN,ICEN)
        END IF

        CALL IMOPEN(IMIOO,'WRITE',' OUTPUT FILE? ','NONAME',NN,MM)
C
        CALL INPUT(' CUTOFF VALUE FOR FILE #2 (0->255)? ',0,255,CVAL)
C
        CALL IMTRAN(IMIOI1,'READ','INTEGER*2',IMA1,512,NN,MM)
        CALL IMTRAN(IMIOI2,'READ','INTEGER*2',IMA2,512,NN,MM)
C
 20     CONTINUE
        DO LX = IRST,IREN
          DO LY = ICST,ICEN
            IF(IMA2(LY,LX).GT.CVAL) IMA1(LY,LX) = 0
```

```
          END DO
        END DO
C
        CALL IMINIT('ERASE')
        CALL IMDISP('WRITE','INTEGER*2',IMA1,512,NN,MM,0,0,'WHITE')
C
        CALL REPLY(' TRY ANOTHER CUTOFF VALUE? ',ANS)
        IF (ANS) THEN
          TYPE *,' OLD CUTOFF VALUE = ',CVAL
          CALL INPUT(' NEW LOWER CUTOFF? ',0,255,CVAL)
          GO TO 20
        END IF
C
        CALL IMTRAN(IMIDO,'WRITE','INTEGER*2',IMA1,512,NN,MM)
C
        END
```

```
C*************************************************
C*************************************************
C
C       INPUT BOUNDED INTEGER FROM TERMINAL
C
        SUBROUTINE INPUT (PROMPT, FIRST, LAST, VALUE)
C
        IMPLICIT NONE
        CHARACTER*(*)  PROMPT
        INTEGER  FIRST, LAST, VALUE
        LOGICAL  VALID
C
1       FORMAT (A)
C
        VALID = .FALSE.
        DO  WHILE (.NOT.VALID)
            TYPE 1, '$'//PROMPT
            READ *, VALUE
            IF (VALUE.LT.FIRST)  THEN
                TYPE *
                TYPE *, 'response can not be smaller than -> ', FIRST
                TYPE *
            ELSE    IF (VALUE.GT.LAST)  THEN
                TYPE *
                TYPE *, 'response can not be larger than -> ', LAST
                TYPE *
            ELSE
                VALID = .TRUE.
            END IF
        END DO
        RETURN
        END
```

```
C       PROGRAM FRECHEN.FOR
C
C       THIS PROGRAM FINDS THE MAGNITUDE OF PROJECTION ON
C       OR THE ANGLE BETWEEN THE NINE DIMENSIONAL SUBIMAGE
C       VECTORS IN AN IMAGE AND THE SUBSPACES DEFINED BY
C       THE USER.
C
        LOGICAL ANG,ANS
        INTEGER*2 IMA1(512,512),IMA2(512,512)
        INTEGER NN,MM,NUM,NPOINT,IRST,ICST,IREN,ICEN
        REAL MSK(9,9),LEN(9),R1,RS,BB,BBW9,BW9,VAR,MAG
C
        MSKIOI = 1
        IMIOI = 2
        IMIOO = 3
C
C       FIND ANGLES OR MAGNITUDE OF PROJECTION
C
        CALL REPLY(' FIND ANGLES (Y/N)? ',ANG)
        IF (ANG)
     &     THEN
              TYPE *,' OUTPUT WILL BE ANGLES!'
           ELSE
              TYPE *,' OUTPUT WILL BE MAGNITUDES!'
              TYPE *
 1            FORMAT(A)
              TYPE 1,'$ ENTER MAX MAGNITUDE = '
              ACCEPT *,MAG
        END IF
C
C       INPUT SUBSPACE VECTORS
C
        CALL SGOPEN(MSKIOI,'READ',' VECTOR FILE ? ','NONAME','REAL',NPOINT)
        CALL SGTRAN(MSKIOI,'READ','REAL',MSK,NPOINT)
        NUM = NPOINT/9
C
C       INPUT IMAGE FILENAMES
C
        CALL IMOPEN(IMIOI,'READ',' INPUT IMAGE FILE? ','NONAME',NN,MM)
C
        CALL IMOPEN(IMIOO,'WRITE',' OUTPUT IMAGE FILE? ','NONAME',NN,MM)
C
        CALL IMTRAN(IMIOI,'READ','INTEGER*2',IMA1,512,NN,MM)
C
C       INPUT DIFFERENT PICTURE SIZES IF DESIRED
C
        IRST = 1
        ICST = 1
        IREN = MM
        ICEN = NN
        TYPE *
```

```
      TYPE *,' NUMBER OF ROWS = ',MM
      TYPE *,' NUMBER OF COLUMNS = ',NN
      CALL REPLY(' IS THE REAL PICTURE SMALLER (Y/N)? ',ANS)
      IF (ANS) THEN
        CALL INPUT(' STARTING ROW? ',1,MM,IRST)
        CALL INPUT(' ENDING ROW? ',IRST,MM,IREN)
        CALL INPUT(' STARTING COLUMN? ',1,NN,ICST)
        CALL INPUT(' ENDING COLUMN? ',ICST,NN,ICEN)
      END IF
C
C     FIND LENGTH OF BASIS VECTORS
C
      DO M1=1,NUM
         LEN(M1) = 0.0
         DO M2=1,9
            LEN(M1) = LEN(M1) + MSK(M2,M1)*MSK(M2,M1)
         END DO
         LEN(M1) = LEN(M1)**0.5
      END DO
C
C     FIRST FIND MAGNITUDE
C
      DO 200 LX = IRST,IREN-2
       DO 100 LY = ICST,ICEN-2
         LX1 = LX + 1
         LX2 = LX + 2
         LY1 = LY + 1
         LY2 = LY + 2
C
         RS = 0.0
C
        DO M1=1,NUM
         R1 = MSK(1,M1)*IMA1(LY,LX) + MSK(2,M1)*IMA1(LY1,LX) +
     &        MSK(3,M1)*IMA1(LY2,LX) + MSK(4,M1)*IMA1(LY,LX1) +
     &        MSK(5,M1)*IMA1(LY1,LX1) + MSK(6,M1)*IMA1(LY2,LX1) +
     &        MSK(7,M1)*IMA1(LY,LX2) + MSK(8,M1)*IMA1(LY1,LX2) +
     &        MSK(9,M1)*IMA1(LY2,LX2)
C
         R1 = R1 / LEN(M1)
         RS = RS + R1 * R1
        END DO
C
        IF (ANG)
     &   THEN
         BB = 0.0
         BW9 = 0.0
         DO L1=0,2,1
          DO L2= 0,2,1
           VAR = IMA1(LY+L2,LX+L1)
           BB = BB + VAR * VAR
           BW9 = BW9 + VAR
          END DO
```

```
            END DO
            BW9 = (BW9*BW9) / 9.0
            BBW9 = BB - BW9
            IF(BBW9.EQ.0)
     &       THEN
              RS = 0.0
             ELSE
              RS = RS / BBW9
            END IF
C
C           EXPAND FRACTION 0->1 TO RANGE 0->255
            IMA2(LY1,LX1) = RS * 255.0
           ELSE
            IF(NUM.EQ.1)
     &       THEN
              RS = 127.5 + R1 * 127.5 / MAG
              IF (RS.LT.0) RS = 0.0
             ELSE
              RS = RS**0.5
              RS = RS * 255.0 / MAG
            END IF
            IF (RS.GT.255.0) RS = 255.0
            IMA2(LY1,LX1) = RS
          END IF
C
 100     CONTINUE
 200    CONTINUE
C
C       DO FOUR THINGS
C        1) COPY 2ND LINE INTO FIRST,
C        2) COPY 2ND TO LAST LINE INTO LAST LINE,
C        3) COPY 2ND COLUMN INTO FIRST,
C        4) COPY 2ND TO LAST COLUMN INTO LAST COLUMN,
C
        DO LY = ICST+1,ICEN-1
           IMA2(LY,IRST) = IMA2(LY,IRST+1)
           IMA2(LY,IREN) = IMA2(LY,IREN-1)
        END DO
C
        DO LX = IRST,IREN
           IMA2(ICST,LX) = IMA2(ICST+1,LX)
           IMA2(ICEN,LX) = IMA2(ICEN-1,LX)
        END DO
C
C       OUTPUT FILE
C
        CALL IMTRAN(IMIOO,'WRITE','INTEGER*2',IMA2,512,NN,MM)
C
        END
```

```
C*************************************************
C*************************************************
C
C       INPUT BOUNDED INTEGER FROM TERMINAL
C
        SUBROUTINE INPUT (PROMPT, FIRST, LAST, VALUE)
C
        IMPLICIT NONE
        CHARACTER*(*)  PROMPT
        INTEGER  FIRST, LAST, VALUE
        LOGICAL  VALID
C
1       FORMAT (A)
C
        VALID = .FALSE.
        DO  WHILE (.NOT.VALID)
            TYPE 1, '$'//PROMPT
            READ *, VALUE
            IF (VALUE.LT.FIRST)  THEN
                TYPE *
                TYPE *, 'response can not be smaller than -> ', FIRST
                TYPE *
            ELSE    IF (VALUE.GT.LAST)  THEN
                TYPE *
                TYPE *, 'response can not be larger than -> ', LAST
                TYPE *
            ELSE
                VALID = .TRUE.
            END IF
        END DO
        RETURN
        END
```

APPENDIX B

This appendix contains the Fortran listings of five short programs used on the Data General NOVA computers of the Electrical Engineering Department of Kansas State University. These programs generate vector files, generate image subareas of ideal edges being rotated and shifted, find magnitude of projection response of edge subareas to vectors and find angle response between edge subareas and vectors.

```
C*************************************************************************
C
C       CRFL
C
C       DB FORTRAN 5 SOURCE FILENAME:           CRFL.FR
C
C       DEPARTMENT OF ELECTRICAL ENGINEERING    KANSAS STATE UNIVERSITY
C
C       REVISION        DATE                    PROGRAMMER
C       --------        ----                    ----------
C       00.0            NOV 15,1982             DANIEL B. SCHOWENGERDT
C       01.0            DEC 31,1982             DANIEL B. SCHOWENGERDT
C
C*************************************************************************
C
        DOUBLE PRECISION DD(50)
        REAL D(50)
C
        ND1 = 0
        NSECT = 0
C
        CALL QUERY('MODIFY EXISTING FILE? ',IANS)
C
        IF (IANS.NE.1) GO TO 100
C
        CALL OPENR(0,'OLD FILENAME? ',4,SIZE)
C
 100    CONTINUE
        CALL OPENW(1,'NEW FILENAME? ',4,SIZE)
C
 200    CONTINUE
        NSECT = NSECT + 1
        WRITE(10,1) NSECT
 1      FORMAT('*** SECTION NUMBER ',I2,' ***')
        TYPE ' '
        ACCEPT 'NUMBER OF DATA VALUES IN THIS SECTION? ',ND
C
        IF (IANS.NE.1) GO TO 550
        DO 400 I400=1,ND
           CALL READR(0,I400+ND1,D(I400),1,IER)
           DD(I400) = DBLE(D(I400))
 400    CONTINUE
 450    CONTINUE
C
        DO 500 I500=1,ND
           TYPE '  D(',I500,') = ',DD(I500)
 500    CONTINUE
C
        CALL QUERY(' CHANGE THESE VALUES Y/N ? ',IANS1)
        IF (IANS1.NE.1) GO TO 650
C
 550    CONTINUE
```

```
      DO 600 I600=1,ND
         WRITE(10,5) I600
 5       FORMAT('   D(',I2,') = ',Z)
         ACCEPT DD(I600)
 600   CONTINUE
       GO TO 450
C
 650   CONTINUE
       DO 700 I700=1,ND
         D(I700) = DD(I700)
         CALL WRITR(1,I700+ND1,D(I700),1,IER)
 700   CONTINUE
       ND1 = ND1 + ND
       CALL QUERY(' ALL SECTIONS DONE? ',IANS2)
       IF (IANS2.NE.1) GO TO 200
C
       END
```

```
C
C       PROGRAM ROEDGE.FR
C
C       THIS PROGRAM CREATES 46 3x3 DISCRETE SUBAREA
C       IMAGES OF AN IDEAL EDGE ROTATING THROUGH
C       46 DEGREES OF ROTATION.  THOROUGHOUT
C       THE ROTATION THE EDGE IS A CONSTANT
C       DISTANCE FROM THE CENTER OF THE SUBAREA.
C
        DOUBLE PRECISION LINE(1806),RAD,INCL
        REAL INIT,IM(9),DENOM
        INTEGER ISHFT,PIX,ST
C
        TYPE ' '
        TYPE ' WELCOME TO THE WONDERFUL WORLD OF EDGES?'
        TYPE ' '
 10     CONTINUE
        ACCEPT ' DISTANCE FROM CENTER OF SUBAREA (0-451)? ',ISH
        IF(ISHFT.LT.0.OR.ISHFT.GT.451) GO TO 10
        CALL OPENW(0,' EDGE FILENAME? ',36,SIZE)
        TYPE ' '
C
C-------INITIALIZE DISCRETE LINE-----------------*
C
C       SET BOUNDARY VALUES
C
        INIT = 903.0
        DO 100 I1 = 1,1806
           IF(I1.GE.1356) INIT = 0.0
           LINE(I1) = INIT
 100    CONTINUE
C
C       FORM DISCRETE LINE
C
        DENOM = 301.0 * 301.0
        DO 290  I1=1,46
           RAD = (DATAN(1.0) / 45.0) * (I1-1)
           IF((I1-1).EQ.0) GO TO 140
           LINE(1355) = 451.0
           INCL = 1.0 / DTAN(RAD)
C
           DO 120 I2=1,451
              LINE(1355-I2) = LINE(1355-I2+1) + INCL
              LINE(1355+I2) = LINE(1355+I2-1) - INCL
              IF(LINE(1355-I2).GT.903.0) GO TO 130
 120       CONTINUE
 130       CONTINUE
C
           IF(LINE(1355-I2).GT.903.0) LINE(1355-I2) = 903.0
           IF(LINE(1355+I2).LT.0.0) LINE(1355+I2) = 0.0
 140       CONTINUE
```

```
C
C-------INTEGRATE-----------------------------------*
C
C
           DO 190 I2 = 1,9
              IM(I2) = 0.0
 190       CONTINUE
C
           ST = 904.0 - ISHFT/DCOS(RAD)
           PIX = ST - 301
C
           DO 270 I2 = 1,3
              PIX = PIX + 301
C
              DO 260 I3 = PIX,PIX + 300
                 IF(LINE(I3)-602.0) 200,200,230
 200             CONTINUE
                 IF (LINE(I3)-301.0) 210,210,220
 210             CONTINUE
                 IM(6+I2) = IM(6+I2) + LINE(I3)
                 GO TO 250
 220             CONTINUE
                 IM(3+I2) = IM(3+I2) + LINE(I3) - 301.0
                 GO TO 240
 230             CONTINUE
                 IM(I2) = IM(I2) + LINE(I3) - 602.0
                 IM(3+I2) = IM(3+I2) + 301.0
 240             CONTINUE
                 IM(6+I2) = IM(6+I2) + 301.0
 250             CONTINUE
 260          CONTINUE
 270       CONTINUE
C
           DO 280 I2 = 1,9
              IM(I2) = IM(I2) / DENOM
 280       CONTINUE
C
           CALL WRITR(0,I1,IM,1,IER)
 290    CONTINUE
        TYPE ' 46 TANTALIZING IMAGES'
        TYPE '       FOR YOUR VIEWING PLEASURE.'
        CALL CLOSE (0,IER)
        END
```

```
C
C       PROGRAM SUBED33.FR
C
C       THIS PROGRAM CREATES 46 3x3 DISCRETE SUBAREA
C       IMAGES OF AN IDEAL EDGE MOVING COMPLETELY
C       THROUGH THE SUBAREA.
C
C
        DOUBLE PRECISION LINE(1806),RAD,INCL
        REAL INIT,IM(9),DENOM
        INTEGER IDEG,PIX,ST
C
        TYPE ' '
        TYPE ' WELCOME TO THE WONDERFUL WORLD OF EDGES?'
        TYPE ' '
 10     CONTINUE
        ACCEPT ' ANGLE OF EDGE (0->45)(I)? ',IDEG
        IF(IDEG.LT.0.OR.IDEG.GT.45) GO TO 10
        CALL OPENW(0,' EDGE FILENAME? ',36,SIZE)
        TYPE ' '
C
C-------INITIALIZE DISCRETE LINE-----------------*
C
C       SET BOUNDARY VALUES
C
        INIT = 903.0
        DO 100 I1 = 1,1806
           IF(I1.GE.1356) INIT = 0.0
           LINE(I1) = INIT
 100    CONTINUE
C
C       FORM DISCRETE LINE
C
        RAD = (DATAN(1.0) / 45.0) * IDEG
        IF(IDEG.EQ.0) GO TO 140
        LINE(1355) = 451.0
        INCL = 1.0 / DTAN(RAD)
C
        DO 120 I1=1,451
           LINE(1355-I1) = LINE(1355-I1+1) + INCL
           LINE(1355+I1) = LINE(1355+I1-1) - INCL
           IF(LINE(1355-I1).GT.903.0) GO TO 130
 120    CONTINUE
 130    CONTINUE
C
        IF(LINE(1355-I1).GT.903.0) LINE(1355-I1) = 903.0
        IF(LINE(1355+I1).LT.0.0) LINE(1355+I1) = 0.0
 140    CONTINUE
C
C-------INTEGRATE--------------------------------*
C
```

```
C
      DENOM = 301.0 * 301.0
C
      DO 290 I1 = 1,46
         DO 190 I2 = 1,9
            IM(I2) = 0.0
 190     CONTINUE
C
         ST = 904.0 - ((FLOAT(I1)-1.0)*10.0)/COS(RAD)
         PIX = ST - 301
C
         DO 270 I2 = 1,3
            PIX = PIX + 301
C
            DO 260 I3 = PIX,PIX + 300
               IF(LINE(I3)-602.0) 200,200,230
 200           CONTINUE
               IF (LINE(I3)-301.0) 210,210,220
 210           CONTINUE
               IM(6+I2) = IM(6+I2) + LINE(I3)
               GO TO 250
 220           CONTINUE
               IM(3+I2) = IM(3+I2) + LINE(I3) - 301.0
               GO TO 240
 230           CONTINUE
               IM(I2) = IM(I2) + LINE(I3) - 602.0
               IM(3+I2) = IM(3+I2) + 301.0
 240           CONTINUE
               IM(6+I2) = IM(6+I2) + 301.0
 250           CONTINUE
 260        CONTINUE
 270     CONTINUE
C
         DO 280 I2 = 1,9
            IM(I2) = IM(I2) / DENOM
 280     CONTINUE
C
         CALL WRITR(0,I1,IM,1,IER)
 290  CONTINUE
      TYPE ' 46 TANTALIZING IMAGES'
      TYPE '       FOR YOUR VIEWING PLEASURE.'
      CALL CLOSE (0,IER)
      END
```

```
C-------------------------------------------------------*
C
C       PROGRAM AMP.FR
C
C       THIS PROGRAM FINDS THE RESPONSE OF THE
C       VECTOR FILE VECTORS TO THE
C       IMAGES IN THE IMAGE FILE.
C       THE CHOICES FOR RESPONSE ARE
C        1) AMP = ((I,V1)**2 + (I,V2)**2)**0.5
C        2) AMP = ABS((I,V1))+ABS((I,V2))
C
C
        INTEGER IO0/0/,IO1/1/,ANS,ICNT
        REAL MSK(9,9),X(9),SIZE,R1,RS,LEN(9)
C
        TYPE ' MASK RESPONSE PROGRAM '
        TYPE ' '
        CALL OPENR(IO0,' VECTOR FILE? ',36,SIZE)
        NUMV = SIZE / 36
C       READ IN MASK VALUES FROM FILE.
C
        DO 150 I1=1,NUMV
           CALL READR(IO0,I1,MSK(1,I1),1,IER)
 150    CONTINUE
        CALL CLOSE (IO0,IER)
C
        CALL QUERY(' SUM OF SQ.(Y) OR ABS.(N) (Y/N)? ',ANS)
        IF(ANS.EQ.1) TYPE ' A = ((I,V1)**2 + (I,V2)**2)**0.5'
        IF(ANS.NE.1) TYPE ' A = ABS((I,V1)) + ABS((I,V2))'
        CALL OPENR(IO0,' IMAGE FILE? ',36,SIZE)
        NUMI = SIZE / 36
        CALL OPENW(IO1,' RESPONSE FILE? ',4,SIZE)
C
C
C       COMPUTE MASK RESPONSE AND COMBINE ACCORDINGLY.
C
        DO 350 I6 = 1,NUMV
           LEN(I6) = MSK(1,I6)*MSK(1,I6) + MSK(2,I6)*MSK(2,I6)
     /               MSK(3,I6)*MSK(3,I6) + MSK(4,I6)*MSK(4,I6)
     /               MSK(5,I6)*MSK(5,I6) + MSK(6,I6)*MSK(6,I6)
     /               MSK(7,I6)*MSK(7,I6) + MSK(8,I6)*MSK(8,I6)
     /               MSK(9,I6)*MSK(9,I6)
           LEN(I6) = LEN(I6)**0.5
 350    CONTINUE
C
C
        DO 500 I5 = 1,NUMI
           CALL READR(IO0,I5,X,1,IER)
           RS = 0.0
           DO 400 I4=1,NUMV
              R1 = MSK(1,I4)*X(1) + MSK(2,I4)*X(2) +
```

```
     /              MSK(3,I4)*X(3) + MSK(4,I4)*X(4) +
     /              MSK(5,I4)*X(5) + MSK(6,I4)*X(6) +
     /              MSK(7,I4)*X(7) + MSK(8,I4)*X(8) +
     /              MSK(9,I4)*X(9)
               R1 = R1/LEN(I4)
               IF(ANS.EQ.1) RS = RS + R1*R1
               IF(ANS.NE.1) RS = RS + ABS(R1)
 400        CONTINUE
            IF(ANS.EQ.1) RS = RS**0.5
C
            CALL WRITR(IO1,I5,RS,1,IER)
 500    CONTINUE
C
        CALL CLOSE(IO0,IER)
        CALL CLOSE(IO1,IER)
        TYPE ' '
        TYPE ' THIS IS THE END '
        END
```

```
C*********************************************************
C
C       FRECHEN
C
C       THIS PROGRAM USES THE DATA FILE CREATED BY
C       THE PROGRAM MEDGE AND COMBINES IT WITH THE
C       THE MASKS INPUTTED BY YOU.
C       IT USES THE FREI AND CHEN ALGORITHM
C       OR JUST GIVES MAGNITUDE RESPONSE.
C
C       EQUATION FOR FREI AND CHEN IS
C       SUM (B,W#)**2/((B,B)-((B,W9)/!!W9!!)**2)
C       NOTE ESPECIALLY THE DENOMINATOR!!!!
C       THIS PROGRAM USES THE DATA FILE CREATED BY
C       THE PROGRAM MEDGE AND COMBINES IT WITH THE
C       MASK INPUTTED BY YOU.
C       IF FINDS THE MAGNITUDE OF THE PROJECTION OF
C       THE IMAGE VECTOR ONTO THE MASK VECTOR.
C       THE MASK VECTOR IS MADE INTO A UNIT VECTOR.
C       PROJECTIONS ONTO SUBSPACES DESCRIBED
C       BY MORE THAN ONE MASK ARE POSSIBLE.
C
C*********************************************************
C
        INTEGER IO0/0/,IO1/1/,ANS,ICNT
        REAL MSK(9,9),X(9),SIZE,R1,RS,SUM,AVE,LEN(9)
        REAL BB,BW9,BBW9
C
        TYPE ' MASK RESPONSE PROGRAM '
        TYPE ' '
        CALL OPENR(IO0,' VECTOR FILE? ',36,SIZE)
        NUMV = SIZE / 36
C       READ IN MASK VALUES FROM FILE.
C
        DO 150 I1=1,NUMV
           CALL READR(IO0,I1,MSK(1,I1),1,IER)
 150    CONTINUE
        CALL CLOSE (IO0,IER)
CC
        CALL QUERY(' ANGLES (Y/N)? ',ANS)
        IF(ANS.EQ.1) TYPE ' RESPONSE IN ANGLES.'
        IF(ANS.NE.1) TYPE ' RESPONSE IN MAGNITUDE.'
        CALL OPENR(IO0,' IMAGE FILE? ',36,SIZE)
        NUMI = SIZE / 36
        CALL OPENW(IO1,' RESPONSE FILE? ',4,SIZE)
C
C
C       COMPUTE MASK RESPONSE AND COMBINE ACCORDINGLY.
C
        DO 350 I6 = 1,NUMV
           LEN(I6) = MSK(1,I6)*MSK(1,I6) + MSK(2,I6)*MSK(2,I6)
```

```
     /                MSK(3,I6)*MSK(3,I6) + MSK(4,I6)*MSK(4,I6)
     /                MSK(5,I6)*MSK(5,I6) + MSK(6,I6)*MSK(6,I6)
     /                MSK(7,I6)*MSK(7,I6) + MSK(8,I6)*MSK(8,I6)
     /                MSK(9,I6)*MSK(9,I6)
          LEN(I6) = LEN(I6)**0.5
 350    CONTINUE
C
C
        DO 500 I5 = 1,NUMI
          CALL READR(I00,I5,X,1,IER)
          RS = 0.0
          DO 400 I4=1,NUMV
            R1 = MSK(1,I4)*X(1) + MSK(2,I4)*X(2) +
     /           MSK(3,I4)*X(3) + MSK(4,I4)*X(4) +
     /           MSK(5,I4)*X(5) + MSK(6,I4)*X(6) +
     /           MSK(7,I4)*X(7) + MSK(8,I4)*X(8) +
     /           MSK(9,I4)*X(9)
            R1 = R1/LEN(I4)
            RS = RS + R1*R1
 400      CONTINUE
          IF(ANS.NE.1) GO TO 450
          BB = X(1)*X(1) + X(2)*X(2) + X(3)*X(3) +
     /         X(4)*X(4) + X(5)*X(5) + X(6)*X(6) +
     /         X(7)*X(7) + X(8)*X(8) + X(9)*X(9)
          BW9 = ( X(1) + X(2) + X(3) +
     /            X(4) + X(5) + X(6) + X(7) +
     /            X(8) + X(9) ) / 3.0
C
          BW9 = BW9 * BW9
C
          BBW9 = BB - BW9
C
          RS = RS / BBW9
C
          GO TO 460
 450      CONTINUE
          RS = RS**0.5
 460      CONTINUE
C
          CALL WRITR(I01,I5,RS,1,IER)
 500    CONTINUE
C
        CALL CLOSE(I00,IER)
        CALL CLOSE(I01,IER)
        TYPE ' '
        TYPE ' THIS IS THE END '
        END
```

REPORT ON THE FAST BOUNDARY DETECTION ALGORITHM OF WERNER FREI AND CHUNG-CHING CHEN

by

DANIEL BENJAMIN SCHOWENGERDT

B. S., Kansas State University, 1979

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1983

ABSTRACT

This masters report deals with edge detection in image processing. This report is a study and discussion of an enhancement/threshold edge detector algorithm presented by Werner Frei and Chung-Ching Chen in their published paper, "Fast Boundary Detection: A Generalization and a New Algorithm". In their paper they present a set of orthogonal functions related to distinctive image features which allows extraction of boundary elements.

This masters report gives a short introduction to the subject of edge detection and then presents and discusses the edge detector algorithm of Frei and Chen. Results are presented which agree with claims, made by Frei and Chen, of improvements over existing techniques. Results of additional testing are also presented, the discussion of which leads to a generalization of image subareas as sums of sinusoid subareas. These sinusoid subareas are then used to explain subarea features such as edges and lines and to explain the edge detection criterion for the algorithm of Frei and Chen.