AN APPROACH FOR MACHINE VISION RECOGNITION OF SEVERELY SKEWED CHARACTERS

By

WILLIAM GLENN PATTERSON II

Bachelor of Science

in Electrical Engineering

Oklahoma State University

Stillwater, Oklahoma

1980

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 1988

AN APPROACH FOR MACHINE VISION

RECOGNITION OF SEVERELY

SKEWED CHARACTERS

Thesis Approved:

C. M. Bass

Thesis Adviser

Richard L. Cummins

Louis G. Johnson

Keith A. Teague

Norman N. Durham

Dean of the Graduate College

PREFACE

The result of this research is the development of a vision system for recognizing severely skewed characters. A series of image enhancement and reconstruction algorithms expand a skewed image in both the x and y directions. The expanded image is recognized and identified by comparison to pre-stored ideal character templates. The vision system reconstructs and recognizes various alphanumeric characters which are tilted with respect to the camera. The data gathered in this research demonstrate that the system's image reconstruction and recognition performance is acceptable for practical use. The only misgiving is the system's poor time efficiency, caused by the great number of repetitious calculations required in many of the algorithms. However, since the algorithms were chosen for adaptation to parallel processing, the outlook for future research and improvements is promising.

I wish to express my sincere gratitude to all of the people who assisted me in my work. In particular, I am indebted to Dr. Charles M. Bacon for his guidance, understanding, and support.

I am also thankful to the other committee members, Dr. Richard L. Cummins, Dr. Louis G. Johnson, and Dr. Keith A. Teague for their help during the course of this work.

My mother, father, brothers, and sisters have my deepest appreciation for their support through the years I was in school. A special note of thanks must go to Karen, my wife, without whom I could not have completed the requirements for this degree.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

LIST OF SYMBOLS

BT - Bottom point of character area (y coordinate)

CCX - Character area center (x coordinate)

CCY - Character area center (y coordinate)

CX(i) - Character area corner (x coordinate)

CY(i) - Character area corner (y coordinate)

EXPA - Expansion ratio for left side

EXPB - Expansion ratio for right side

LS - Left point of character area (x coordinate)

M(i) - Card diagonal and quadrant angles

P(i) - Side angles

QX(i) - Quadrant center line (x coordinate)

QQX(i) - Quadrant/character area intersect (x coordinate)

QQY(i) - Quadrant/character area intersect (y coordinate)

RA(i) - Left side character area length

RB(i) - Right side character area length

RS - Right point of character area (x coordinate)

SX(i) - Card corner (x coordinate)

SY(i) - Card corner (y coordinate)

X(i) - Card scan intersect point (x coordinate)

Y(i) - Card scan intersect point (y coordinate)

CHAPTER I

INTRODUCTION

The use of digital computers to process image data is not a new concept, but the use of image processing in industry is still in its infancy. In the past, vision systems required extensive processing power, but with the advances in digital computer technology, a low cost, efficient vision system is a reality. Several factors are responsible for the rapid evolution of image processing: the greater dynamic range of the vision system versus the human eye, the presentation of a larger amount of information in a compact form, and the capability to manipulate the image data for specific needs. The purpose of this research is to develop an integrated vision system that can recognize severely skewed character images.

All image processing systems are made up of a camera connected to a computer which is controlled by the software running in it. The camera records the image data, which is a two-dimensional representation of a three-dimensional scene. The camera forms this two-dimensional image by digitizing the received light intensities over the entire image. The digitized information representing the light intensities of the image at each pixel location is known as the digital

image. The digital image is usually in one of two formats, either in a binary or a gray scale representation. The binary representation uses only two levels to describe the light intensities, producing a black and white image. The gray scale representation uses a range of values corresponding to the light intensities at each location, producing incremental gray levels that range from absolute black to pure white.

There are several factors which must be overcome by the vision system to enhance its performance and insure its overall reliability. The first factor is the system's ability to distinguish the object of interest from the background information. Another factor is the system's ability to compensate for poor image quality. Poor image quality may be caused by effects such as inadequate lighting, shallow depth of focus, and movement of the object during the image capturing process. The next limiting factor is the vision system's ability to accurately recognize the character, object, or pattern in the image. This factor influences the possible responses of the vision system, whether it is as simple as writing a message to a terminal or as complex as the detection and tracking of infrared images in outer space. The final factor is the amount of data to be processed, which can affect the response times in real-time applications. This impact on system performance can be reduced by implementing a parallel processing scheme to enhance data throughputs.

Previous research in character recognition is reviewed in Chapter II. The approach used in this research is presented in Chapter III. The data gathering techniques, as well as the data gathered, are presented in Chapter IV. Chapter V discusses the possible enhancements to the vision system by the implementation of a parallel processing system to improve the system time performance. The results in Chapters IV and V are the basis for the comments and recommendations in Chapter VI for future research into image processing for use in industry.

CHAPTER II

BACKGROUND

Computer vision systems have a wide variety of applications, ranging from reading labels to tracking infrared images in low earth orbit. These applications may have unique features and problems, but they share the same basic concepts. These concepts, collectively referred to as the image development process, are outlined in Figure 1.

Image Development Process

The image development process begins by capturing an image using a linear array of photosensitive elements to measure the intensity of the incoming light. The light intensity measurements, in the form of analog voltages, correspond to the amount of light energy received during a specified time frame. These voltages are converted to discrete gray scale levels for each pixel location in the camera's field of view. The gray scale levels fall between the black and white reference levels discussed in Chapter I. After the gray scale information is stored in matrix form, the digital image is ready for processing and recognition by the vision system. Additional information on capturing images with CCD arrays and the digitization process can be

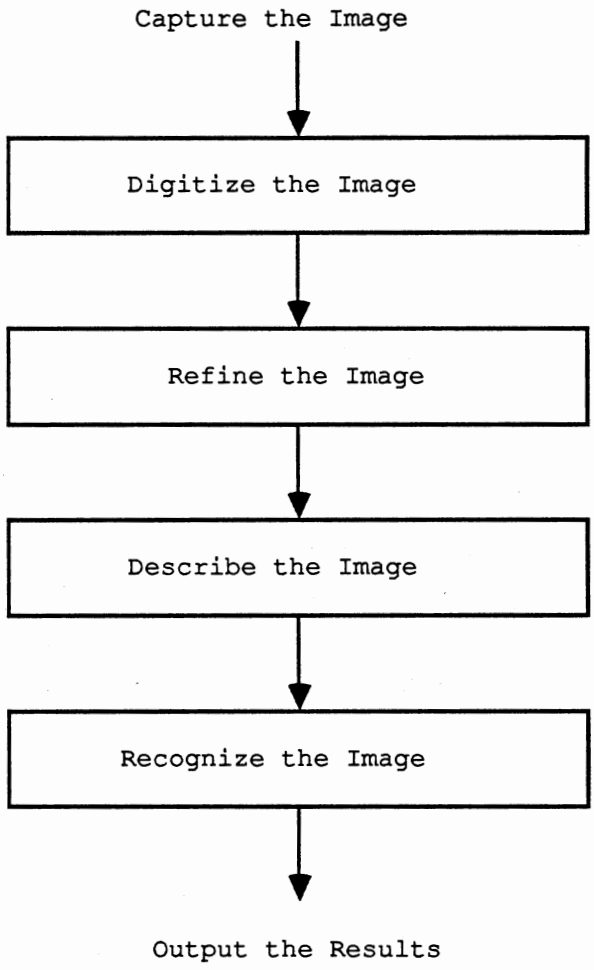


Figure 1. Image Development Process

found in articles by Bower (1) and Amelio (2).

A main goal of image processing is to produce a high quality image to facilitate image recognition. To achieve the higher quality, the image is refined using either enhancement or reconstruction techniques. The distinction is that image enhancement improves the quality, such as clarity, of the captured image, while image reconstruction corrects distortions, such as rotation and translation. Although

image enhancement procedures may be applied to any captured image, image reconstruction requires some a priori knowledge of the system to adequately restore the image to a recognizable form.

A second goal of image processing is to establish a descriptor which can uniquely identify the object, character, or pattern of interest within an image. These descriptors fall into two categories: image segmentation and regional descriptors. Segmentation partitions the image into regions of interest, such as detecting the edges of an object. The regional descriptor characterizes the various segmented regions as to their type, shape, orientation, etc.

Once the descriptor is established, the vision system interprets the image by comparing the descriptor with pre-stored information. The simplest approach is to compare the object with pre-stored models, known as templates, on a pixel by pixel basis. A second approach compares a regional descriptor of the overall shape of the image structures, such as lines, corners, and curves, to pre-stored data.

Concept Development

The initial interest in the development of a vision system recognition process for severely skewed characters began as a portion of a research project undertaken by the School of Electrical and Computer Engineering for the Boeing Military Airplane Company in Wichita, Kansas. Boeing is developing an automated wire harness assembly system known as

the Large Robotically Enabled Assembly of Cable Harnesses (REACH) Cam II project. The specific vision system application is to inspect and verify wire harness parts by reading alphanumeric characters on the labels for each part. The orientation of the labels with respect to the camera is unknown, hence requiring a method to reconstruct and read the character images from skewed labels.

One solution to the character recognition problem is to use robotics to move the camera or the label to a known orientation before capturing the image. A second situation is to assume that the camera position is fixed and the labels are read as they appear. This research concentrates on the second of these situations. To simulate the alphanumeric characters on the labels, a square card with a single character on it was selected.

Literature Review

The following discussion summarizes a literature review investigating various methods to develop the proposed character recognition system.

Image Refinement Techniques

The image enhancement and image reconstruction processes mentioned above are discussed more fully in the following sections.

Image Enhancement

There are two basic categories of enhancement techniques. The first includes frequency-domain methods and the second includes spatial-domain methods. The frequency methods modify the frequency representation of the image, while spatial methods alter the image itself by direct manipulation of the pixels.

Frequency domain techniques begin with the concept of the Fourier transform, which is the frequency description of the digital image f(x,y). Gonzalez (3; pp. 36-38) presents the basics of the Fourier transform, which is written as

$$\mathcal{F}\{f(x,y)\} = F(u,v) = \iint f(x,y)\exp[-j2\Theta(ux+xy)]dxdy \quad (2.1)$$

Another important concept is that of the convolution of two signals, which results in a new function g(x,y), also covered by Gonzalez (3: p. 65). This is given by

$$g(x,y) = f(x,y)\bullet h(x,y) = \iint f(u,v)h(x-u,y-v)dudv \quad (2.2)$$

The above integral, known as the convolution integral, is the sum of the products of the signals for all points. The computation of this integral is difficult, but combining it with the Fourier transform can produce very useful results. The Fourier transform pair for the convolution of two signals is as follows:

$$G(u,v) = \mathcal{F}\{g(x,y)\} = \mathcal{F}\{f(x,y)\bullet h(x,y)\} = F(u,v)*H(u,v) \quad (2.3)$$

The significance of G(u,v) is that it simplifies the analysis by using a simple multiplication in the frequency-domain as opposed to an integral calculation in the time-domain. The above discussion applies to the continuous time case, but the methods can be extended to discrete time applications. These concepts, as well as those of sampling theory and the Fast Fourier Transform, are interesting topics for further investigation, but are not covered in this research. The material is covered quite well in Oppenheim (4).

Spatial techniques use transformations to enhance the image. The type of transform used depends on the criteria chosen, such as contrasts between image features, noisy data, and gray scale levels. The remainder of this section deals with specific applications of both frequency and spatial enhancement techniques.

Gray Scale Histograms. A gray scale histogram represents the statistical distribution of digitized intensities of an image; more simply, it is a record of the number of pixels at each gray level. Projecting each gray level in the input image through a transformation curve gives a new representation of the image. The result is an increase in the range of the gray levels in the output image. Unfortunately, the use of a linear transform causes some blurring of the resulting image at its edges. A linear transform is shown in Figure 2 (5; p. 61). To more fully utilize the output gray levels, a piecewise linear

approximation can be used. The advantages of using this approach are more control over image contrast and a reduction of histogram asymmetry. Figure 3 depicts a typical piecewise linear transform (5; p.64).

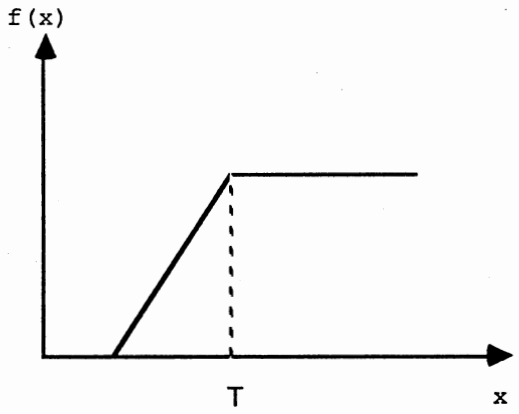


Figure 2. Linear Transform

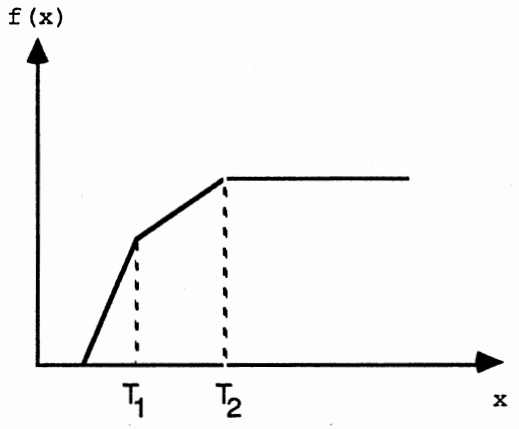


Figure 3. Piecewise Linear Transform

A more general nonlinear transformation, called a histogram equalization, uses a probability distribution function (PDF). The specifics of the PDF can be found, for example, in Papoulis (6). In general, it is a normalized measure of the total number of pixels in the histogram between zero and each gray level. Using the PDF as a transform produces an image of approximately uniform density as seen in Figure 4 (5; p. 64).

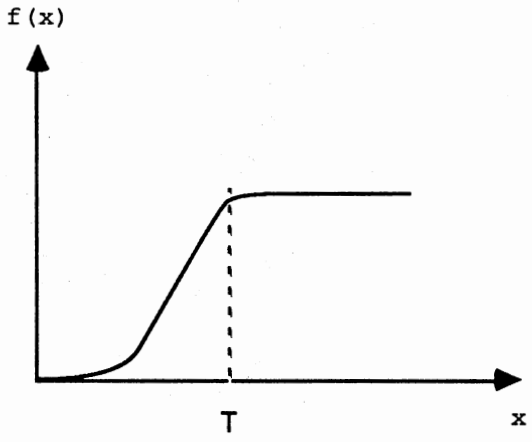


Figure 4. Nonlinear Transform

Thresholding. Thresholding enhances the contrast of an image by darkening the intensity of the output image for input intensities below a specified value and brightening the image for the intensities above that value. The result is a

compression of the gray levels toward each end of the spectrum. Figure 5 shows the thresholding transform (5). A practical application of thresholding is to enhance the output of the histogram transformation techniques. Another application of thresholding is to detect differences between two similar images. However, in any application of thresholding, the problem of solving for the optimal threshold value must be solved.

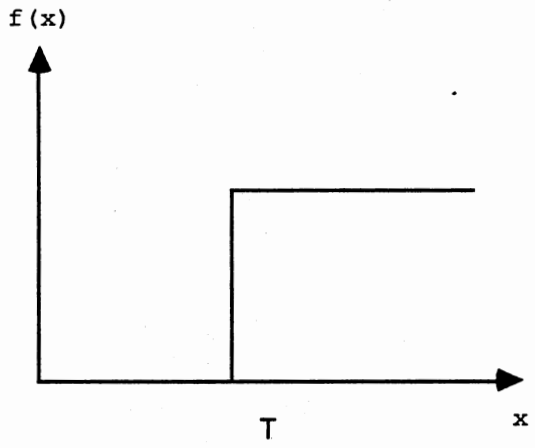


Figure 5. Threshold Transform

Image Smoothing. Image smoothing is similar to the preceding transforms in that it also performs a pixel by pixel transformation. However, image smoothing alters the gray level of a pixel according to its relationship with its neighbors. There are two general categories of image

smoothing techniques: neighborhood averaging and filtering of undesirable characteristics.

Neighborhood averaging is a simple technique which alters the image f(x,y) to produce a smoothed image g(x,y). The traditional definition of a neighborhood is the points that lie inside a circle of radius R. This radius can vary depending on the specific application. Two typical examples are shown in Figure 6 (3; p. 137). The image resulting from neighborhood averaging is smoothed, though somewhat blurred. One solution to the blurring problem is to use the thresholding technique as given by

$$g(x,y) = \begin{cases} \frac{1}{M} \sum_m \sum_n f(m,n) & \text{if } |f(x,y) - g(x,y)| > T \\ f(x,y) & \text{otherwise} \end{cases} \qquad (2.4)$$

where M is the number of points in the neighborhood and T is the threshold value.

An article by Ann Scher et. al. (7; pp. 153-158) describes other types of neighborhood averaging techniques. The first, referred to as half-neighborhood averaging, calculates an average with only those neighbors whose gray levels are closest to the point in question. The method produces a smoother image with sharper borders. The second neighborhood averaging technique uses weighting factors as a measure of the confidence that the neighbor belongs to the same region as the center point. The weighted averaging method improves the image quality, yet it still has some blurring at the edges.

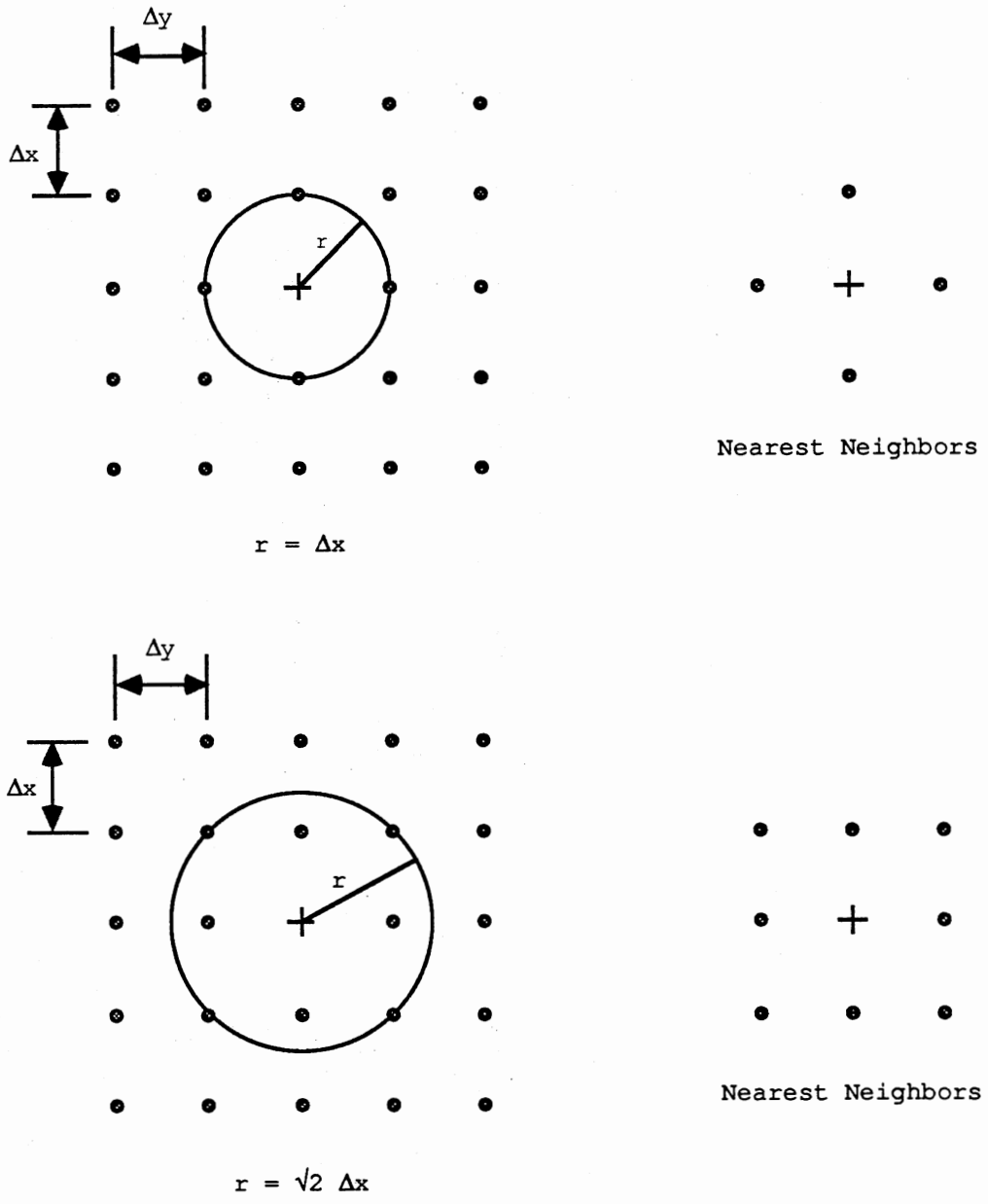


Figure 6. Neighborhood Averaging

The second general category of image smoothing is that of frequency-domain filtering. When scanning an image, the rate of change in the gray levels determines the frequency components present. Low frequencies correspond to smooth regions of nearly uniform gray levels, while high frequency components are the abruptly changing gray level at an edge or other sharp transition. Low pass filtering reduces image sharpness by suppressing the high frequency components. Image filtering uses convolution and Fourier transformations to obtain the desired filtered image. The convolution of the original transformed image F(u,v) and the filter H(u,v) produces a frequency representation of the filtered image G(u,v). Taking the inverse Fourier transform gives the desired smoothed image g(x,y) as shown:

$$g(x,y) = \mathcal{F}^{-1}\{G(u,v)\} = \mathcal{F}^{-1}\{F(u,v) \cdot H(u,v)\} \qquad (2.5)$$

Several common low pass filter equations and frequency responses are shown in Table I, Low Pass Filter Equations and Figure 7, Low Pass Filters (3; pp. 140-150).

Image Sharpening. Image sharpening is a critical process when distinguishing the character's edges is of primary interest. The techniques described below highlight the edges and thereby improve the image quality. These techniques utilize both the spatial and frequency domains.

The first technique uses differentiation by means of a gradient to sharpen the the image. Since the rate of change

TABLE I

LOW PASS FILTER EQUATIONS

Filter Type	Equation
Ideal	$H(u,v) = \begin{cases} 0 & \text{if } [D(u^2+v^2)]^{1/2} \le D_o \\ 0 & \text{if } [D(u^2+v^2)]^{1/2} > D_o \end{cases}$
Butterworth	$H(u,v) = \dfrac{1}{1 + 0.414 * [D(u,v)/D_o]^{2n}}$
Exponential	$H(u,v) = \exp[-(D(u,v)/D_o]^n$
Trapezodial	$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_o \\ \dfrac{D(u,v) - D_o}{D_o - D_1} & \text{if } D_o < D(u,v) \le D_1 \\ 0 & \text{if } D(u,v) > D_1 \end{cases}$

of the gray levels is highest at an object's edges, then the differential is greatest at the object's boundary. The result is that an image with distinct edges may reduce to a line drawing of the original image (8; pp. 298-309).

The other technique suppresses the low frequency components by high pass filtering. The high pass equivalents to the low pass filters previously discussed are given in Table II and are shown in Figure 8 (3; pp.161-164). These filters severely attenuate the low frequencies, making smooth gray levels appear the same. To remedy this, the gray levels

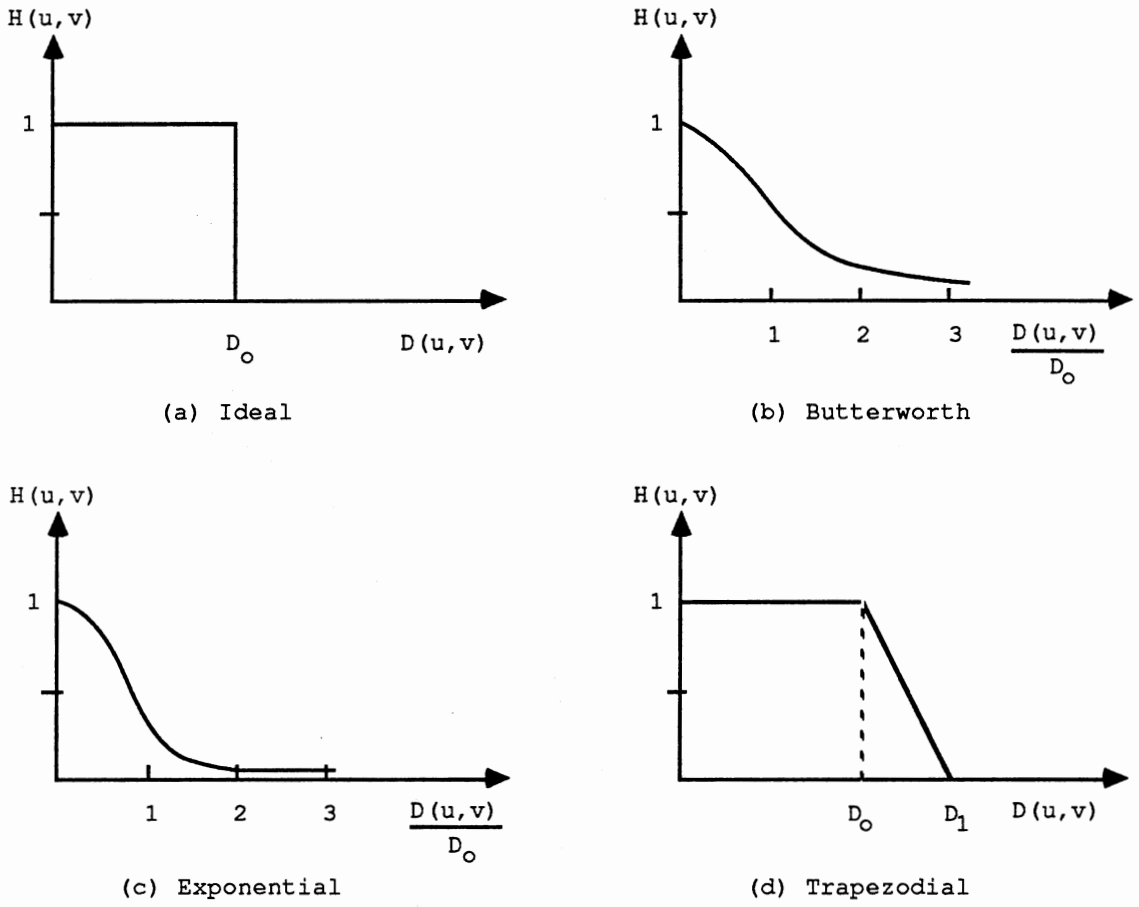


Figure 7. Low Pass Filters

can be redistributed using a histogram equalization technique, which improves the image quality by preserving some of the low frequency components.

Image Reconstruction

As in image enhancement, the purpose of image reconstruction is to improve the image quality. The choice of reconstruction technique depends on what is known about

TABLE II

HIGH PASS FILTER EQUATIONS

Filter Type	Equation
Ideal	$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \leq D_o \\ 0 & \text{if } D(u,v) > D_o \end{cases}$
Butterworth	$H(u,v) = \dfrac{1}{1 + 0.414 * [D_o / D(u,v)]^{2n}}$
Exponential	$H(u,v) = \exp[- 0.347 * (D_o / D(u,v))^n]$
Trapezodial	$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) > D_o \\ \dfrac{D(u,v) - D_o}{D_o - D_1} & \text{if } D_o \geq D(u,v) > D_1 \\ 0 & \text{if } D(u,v) \leq D_1 \end{cases}$

the vision system being developed. The two basic types of reconstruction algorithms are geometric transforms and inverse filter transforms. The geometric transform modifies the image spatial information, such as its shape, size, and orientation. On the other hand, inverse filtering uses a filter function to compensate for image degradation, resulting in a better representation of the original image. The following is a brief description of several reconstruction techniques examined for possible use in this research.

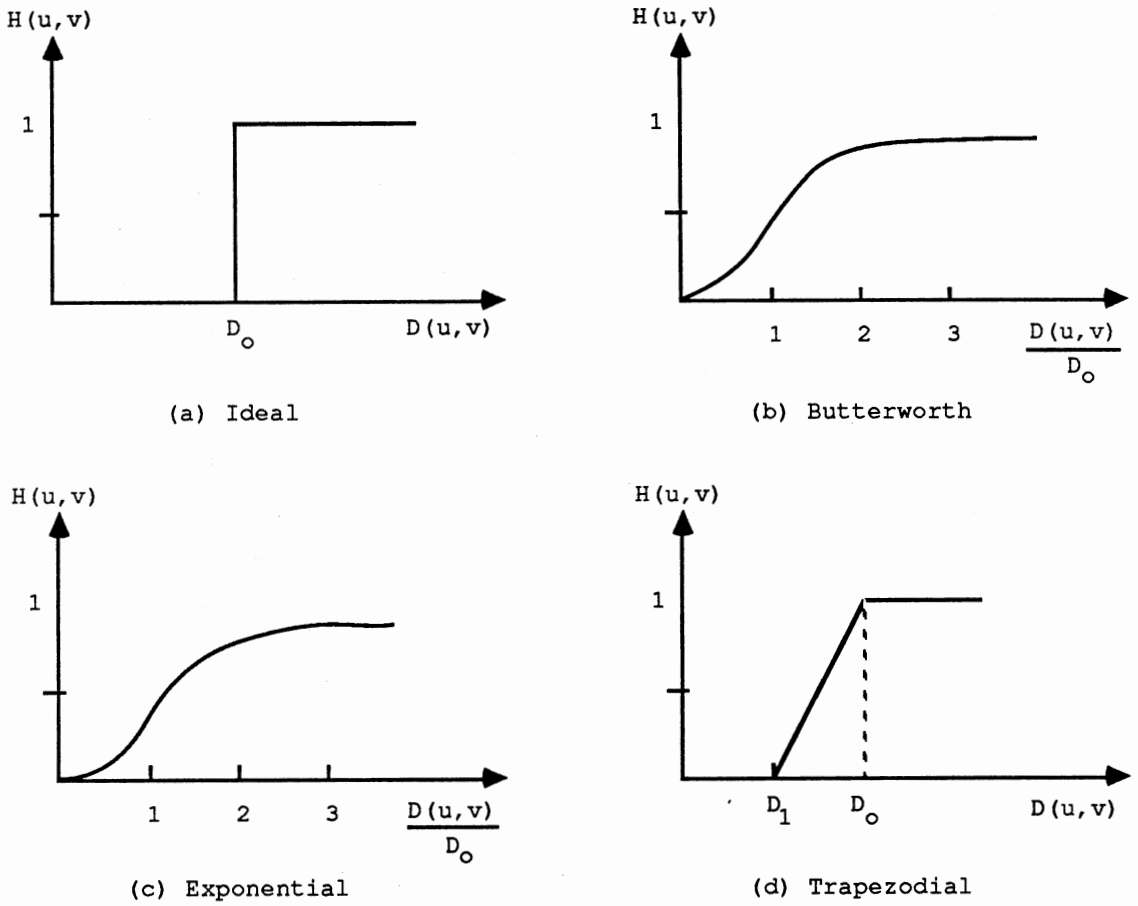


Figure 8. High Pass Filters

Geometric Transforms. Modifying the spatial information, as mentioned above, can be a simple image translation or a linear expansion, as well as a complex coordinate transformation. The basis for geometric transforms is to define the relationship between the pixels in the input and the output images. However, a mapping problem can arise with this type of transformation. The calculated output pixel values do not always correspond to whole number pixel locations. Hence, some form of

interpolation is required to map the input image's pixel intensity to the appropriate transformed image pixel location. The easiest solution is to assign the intensity to the nearest pixel location, but this can lead to significant rounding errors.

The coordinate transformation process becomes necessary when it is desired to rotate an object or an image to a standard orientation. Given a coordinate system, x_1 vs. y_1, such as the one shown in Figure 9, corresponding points in a new coordinate system, x_2 vs. y_2, can be calculated. The only requirement of the new coordinate system is that its origin be the same as that of the original coordinate system. Assuming an angle of rotation of θ between the two coordinate systems, the trigonometric relationship between them is given by (3; p.109)

$$x_2 = x_1 * \cos \theta + y_1 * \sin \theta \qquad (2.6)$$

$$y_2 = y_1 * \cos \theta - x_1 * \sin \theta \qquad (2.7)$$

where (x_1,y_1) corresponds to the old coordinate position and (x_2,y_2) is the new coordinate point.

Inverse Filtering Transforms. Inverse filtering is predominantly a frequency-domain technique, which attempts to correct irregularities introduced during image capturing. Inverse filtering has become popular in part due to the Fast Fourier Transform (FFT) algorithm. The FFT can quickly transform a large two-dimensional image into a discrete

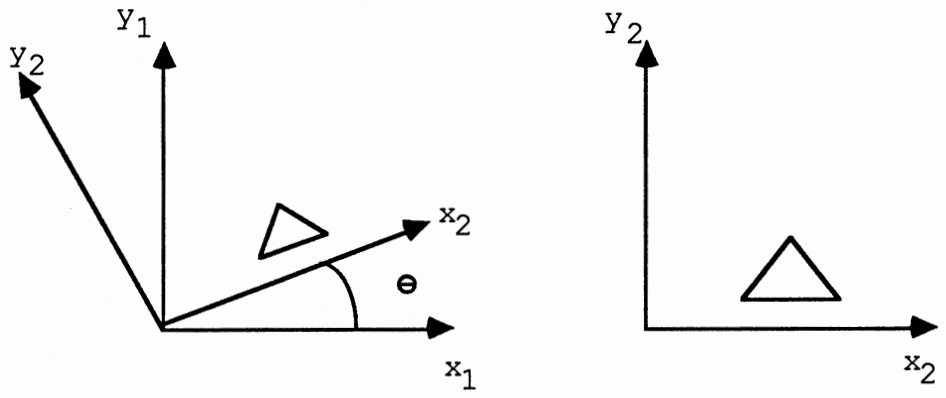


Figure 9. Coordinate Transformation

frequency-domain representation of that image. The only drawbacks to the FFT algorithm are that (1) the resolution is often insufficient and (2) false details due to sidelobes may be present.

The relationship between the captured image, g(x,y), and the ideal original image, f(x,y), is given by (3; p. 184)

$$g(x,y) = h(x,y) \bullet f(x,y) + n(x,y) \tag{2.8}$$

where h(x,y) is the filter function describing the irregularities to be corrected and n(x,y) is any additional noise present. Using the Fourier transform methods and solving for the ideal image gives the equation

$$F(u,v) = \frac{G(u,v)}{H(u,v)} - \frac{N(u,v)}{H(u,v)} \tag{2.9}$$

Then, the inverse FFT of F(u,v) yields the spatial-domain solution for the ideal original image, f(x,y).

One practical application of inverse filtering is to restore an image that was blurred by some constant motion during the image capturing process. For instance, in the case of an object moving on a conveyor belt past a stationary camera, the conveyor belt speed can be assumed constant. Hence, the function h(x,y) represents this motion. The noise n(x,y) might be due to variations in lighting. With this information available, it is easy to see how the resulting filtered image can be obtained. However, the actual transform functions h(x,y) and n(x,y) can be difficult to model, resulting in a less than ideal image.

Image Description Techniques

The image description processes mentioned previously are discussed more fully below.

Segmentation

One approach to image segmentation is the use of edge detection algorithms, in which the image is analyzed on a pixel by pixel basis. The first of these techniques uses gray level thresholding to detect regions or boundary points such that a level change corresponds to the intersection of the background and the object. As the image is scanned line by line, the thresholding indicates a transition from one state to the other, denoting the presence of a boundary. A problem with this technique is setting the threshold value to

obtain the best results. If the number of images to be analyzed is small, then the thresholds can be set using trial and error. If the number of images is large, however, an automated method of determining the threshold is required. One possible method utilizes a minimum mean square estimation or similar technique to determine the optimal threshold values.

Another edge detection technique uses a gradient operator to find the transition from one region to another. Although there are many types of operators, the discussion here is limited to the Sobel operators. Using an operator to calculate the gradient is similar to passing a window function or template over the image. Figure 10 is a 3 x 3 pixel region of the image. Using the Sobel operators shown in Figure 11 (3; p. 338 and 9; pp. 242-244), the gradients in the x and y directions are given by

$$G_x = (g + 2*h + i) - (a + 2*b + c) \qquad (2.10)$$

$$G_y = (c + 2*f + i) - (a + 2*d + g) \qquad (2.11)$$

where the variables a-i are the intensities of the image at that pixel location. Treating G_x and G_y as components of a vector, it is easy to calculate the magnitude of the vector as

$$G = [\, G_x^2 + G_y^2 \,]^{0.5} \qquad (2.12)$$

and the angle between them by

$$\Theta = \arctan (G_y / G_x) \qquad (2.13)$$

Moving these operators around the image forms the description of the image.

a	b	c
d	e	f
g	h	i

Figure 10. 3 x 3 Image

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Figure 11. Sobel Operators

A second approach to the image segmentation process is a regional technique, as opposed to the edge detection techniques. One such technique is region growing, which divides the image into regions of similar characteristics by

first locating a single point in each region. Next, the region-growing algorithm examines the neighboring points for similar characteristics. Appending only those neighbors which are similar and discarding those that are different, the regions grow from a single point. This process continues until all of the pixels are enclosed in the regions. Unfortunately, the initial point in the region is seldom known; therefore, this technique has limited usefulness.

Regional Descriptions

A regional description represents the characteristics of a region, yet is not severely affected by such things as rotation, translation, etc. The descriptor chosen brings out those features which uniquely classify the various regions. Some typical descriptors are the Fourier descriptor, a physical description of the properties of a region, and the use of central moments (3).

The Fourier descriptor uses the discrete Fourier transform to extract information from the boundary points of a specific region. This information is then used to classify that region by its shape, size, orientation, etc. The use of the discrete Fourier transform alone is not enough to fully characterize the region under all circumstances. To obtain the desired set of characteristics, the extracted data is manipulated in the frequency domain, which allows the system to control the image size using the linearity properties of

the Fourier transform. This property allows the multiplication by a constant in frequency-domain to correspond to a multiplication by the same constant in the spatial domain. Similarly, the rotation can be controlled by multiplying each coordinate by $e^{j\theta}$ where θ is the angle of rotation. The result of these and other manipulations is the establishment of a standard size, orientation, and starting point. The reader is referred to Gonzalez for further investigation of this subject.

The second type of descriptor describes the physical aspects of the image. Typically the physical descriptors are such things as the number of holes in a region, the number of connected points in a region, etc. These descriptors can be combined to form a standard for comparing the images to pre-stored descriptors.

The last type of descriptor to be discussed here is the use of central moments. Given a two-dimensional image, the central moments are given by

$$\mu_{pq} = \sum \sum (x - x)^p (y - y) f(x,y) \qquad (2.14)$$

Further derivations yield the second and third moments as well as a set of invariant moments; however, the details of these derivations are outside the scope of this text. These invariant equations have been shown to be invariant to translation, rotation, etc.

Image Recognition Techniques

The interpretation of the image or its descriptor is the ultimate goal of a vision system. Two image recognition schemes that are widely used are template matching and relational characteristic matching. Template matching compares the restored image with pre-stored image, known as templates, while the relational characteristic method uses regional descriptors, to identify the object, character, or pattern within the image. Additional information on these topics can be found in Gostasby (10; pp. 374-378) and Gonzalez.

Template Matching

In template matching, the template acts as a window as it is shifted across the restored image. As the template passes across the image, the amount of correlation between the image and the templates is calculated. The correlation calculation chosen depends on the accuracy required and the computational time needed.

One template matching technique compares the absolute difference between the image and the template on a pixel by pixel basis. By calculating the sum of the pixel to pixel correlation, the measure of correlation between the two images is attained. The highest point of correlation identifies the object and its position. This method can be implemented fairly quickly in many applications, but it is

also highly susceptible to errors due to variations in size, rotation, and translation of the image. The absolute difference is given by

$$g(x,y) = \sum \sum \mid f(x,y) - h(x,y) \mid \qquad (2.15)$$

where $h(x,y)$ and $f(x,y)$ are pixel locations in the image and the template.

A second template matching technique uses the Fourier transform concept to calculate a correlation coefficient between the image and the templates. The calculation of the correlation coefficient is given by the convolution of the image and the template. Using the FFT algorithms discussed earlier, the convolution of the image and the template is given by

$$G(u,v) = \sum \sum F(u,v) \cdot H(u,v) \qquad (2.16)$$

Then analyzing the correlation coefficients, the object and its location are found.

Characteristic Matching

In characteristic matching, the results of the regional descriptors are organized as to relationships between the various regions identified. The structure of these relationships can be described using coded structures, known as grammars, which classify the regions in the image by its size, shape, appearance, etc. Once the grammar is

established, the vision system can classify and identify the object by comparing the relationships with pre-stored data.

Evaluation of Alternative Approaches

The above-mentioned techniques are but a brief description of the methods that could be implemented for the proposed vision system. To determine the specific approach for this research, several factors were examine to determine their impact on the proposed vision system. Number one was the equipment that was available at the onset of this research. This equipment includes a VAX 11/750 used for the data processing and the COMTAL Image Processing System used to capture and display the images. The next factor, prior to the selection of specific image processing algorithms, was the computer language. Due to a strong background in FORTRAN, it was the language of choice.

The literature review helped to narrow down the choice of algorithms. Each algorithm was examined for ease of operation and adaptability to parallel processing. The first conclusion drawn from the literature review was that a binary image representation was easier to work with than the gray scale representation. Hence, the threshold filter, discussed in Chapter II, was needed. The image recognition technique was the next major choice. For simplicity, template matching was chosen. Template matching requires either the construction of numerous templates at various orientations, or, as is used, the reconstruction of the captured image for

comparison to a single set of ideal character templates. This necessitated the use of the image reconstruction algorithms. For the reconstruction, a series of geometric transforms was chosen over inverse filtering because inverse filtering requires a more precise knowledge of the physical system.

The final factors analyzed were the assumptions required to standardize the recognition process. First, the entire card or label must be contained within the boundaries of the camera's field of view, or the captured image. Second, the edges of the white card must be easily distinguishable.

CHAPTER III

EXPANSION TECHNIQUE

The following sections present the approach chosen for vision system recognition of severely skewed characters. First, a top down description of the functional flow is presented. Then, a detailed description of the specific algorithms used are presented. The algorithms for the process were chosen from the techniques and the factors outlined in Chapter II.

Functional Flow Description

The vision system approach to character recognition can be broken down into four basic functions as shown in Figure 12. The first of these functional blocks is the parameter initialization function, which initializes the system level parameters, reads in the captured image, reads in the templates, and runs the initial threshold filter. The threshold value is set so as to accentuate the edges of the card against the background.

The initialization process is followed by the calculation of the card's parameters. These calculations include a search algorithm to find an x and y coordinate pair for each side of the card, as well as the slope of each side. Using

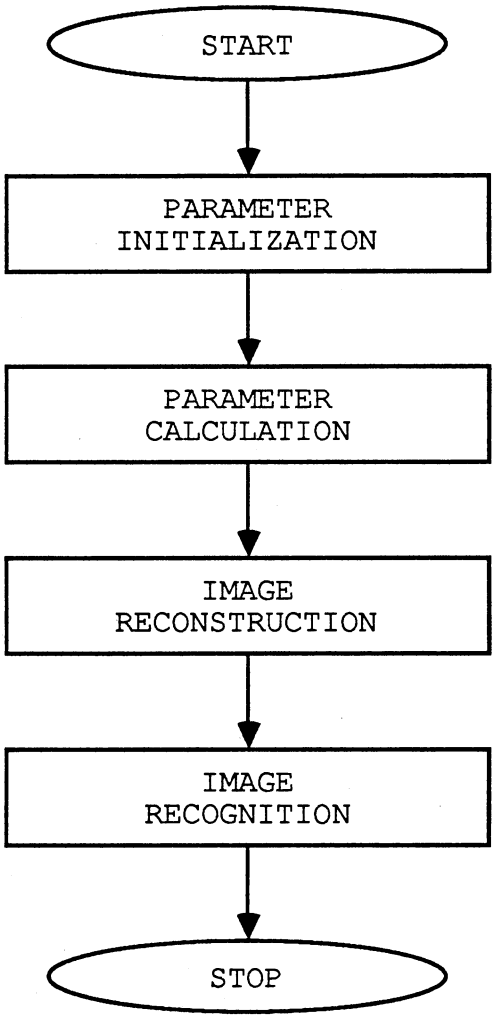


Figure 12. Top Level Functional Flow

these coordinates and slopes, the x and y coordinate pairs of the card's four corners are calculated. Further geometric calculations define the character area, which is a window on the card where the character is known to reside.

The reconstruction process begins by applying a second threshold filter, which accentuates the character on the card. Then, if needed, a rotational correction is performed.

Prior to any further processing, the character area window is centered with respect to the character. This is followed by the expansion of the image first in the x direction, then in the y direction.

The recognition process uses a difference algorithm to calculate a correlation coefficient for the image with respect to each template. The template with the highest correlation is determined to be the recognized character. Then the results of the simulation run are saved for post-processing analysis.

Detailed Software Description

The software which implements the above-mentioned approach for vision system recognition is an automated routine, processing skewed image data with minimal operator interaction. To control the recognition process, an executive routine calls a series of FORTRAN subroutines to perform the various functions. The following discussion describes the operation of each software module. The structure is outline in Appendix A in flow chart form.

Executive Routine

The executive routine, or executive, performs many functions. First, it initializes a series of common data blocks and the global parameters used by the subroutines. The routine then calls up two subroutines which read in the

character templates and the captured input image.

In order to simultaneously display the results of each step of the image processing in a 512 x 512 pixel area, the executive develops a smaller representation of the captured image in its original form (and also later, after it has been thresholded and rotated). These 512 x 512 images are reduced by assigning the data from every even-numbered pixel to a 256 x 256 matrix. The remaining results (the x-expanded image, the y-expanded image, the translated image, and the template chosen by the program as the recognized character) are all 128 x 128 pixel representations. Thus, the seven stages of the image processing mentioned above can be displayed in a 512 x 512 pixel area, the "display matrix."

After reducing the original captured image to 256 x 256 matrix, as just described, the executive routine calls subroutines to calculate the card parameters, such as the side coordinates, and the character area window. The results of these calculations are written to a data file for post-processing analysis. The program then calls the character threshold routine, which produces the thresholded image to be reconstructed. At this point, the 256 x 256 pixel representation of the thresholded image is developed for the display matrix, as described above. This ends the preprocessing portion of the executive routine.

In the reconstruction phase, the executive calls the rotate, translate, and expansion algorithms to perform the actual image reconstruction. The 256 x 256 pixel

representation of the rotated/translated image and the 128 x 128 pixel x-expanded and y-expanded images are assigned the appropriate portions of the 512 x 512 display matrix. In addition, the card parameters modified by the rotation and translation algorithms are written to a data file for later analysis.

At this point, the executive enters the recognition phase. The first subroutine called is a segmentation algorithm which examines the image to calculate character features, such as height and width. This information is used to center the image within the character area and to define the boundaries for the template matching recognition subroutine, which the executive routine calls up next. The results of the recognition are output to the terminal and written to a data file. The remaining portions of the 512 x 512 pixel display matrix are filled in with the translated image and the template of the character recognized by the vision system.

The final steps in the executive program are to convert the display matrix from an integer format to a byte format, then to save this matrix to disk memory for transfer to the COMTAL system for display.

Read in the Templates

The subroutine INPUT.FOR reads in the templates from pre-stored byte-format files. Once all the templates are read

in, they are converted from a byte format to an integer format using the FORTRAN equivalence operator. The integer-format templates are stored in a 128 x 128 x 6 matrix, TMP(i,j,k). The template matrix is passed back to the executive for later processing.

Read in the Image

The subroutine IMGREAD.FOR prompts the operator to input the VAX file name of the saved image to be processed. After the image is read into memory, the program prompts the operator for the initial threshold value. This threshold filter differentiates the white area of the card from the black background. The input image is thresholded after it is converted to integer format and saved in the input image matrix f(x,y). The original gray scale image is saved as IMAGE_IN(x,y). These arrays are passed back to the executive for further processing.

Find the Card's Sides

The subroutine SIDES.FOR is called up to find a point on each of the card's four sides, then calculate the angles corresponding to those sides and the orientation of the card in the image plane. To obtain this information, the program scans the image horizontally pixel by pixel, starting at position (1,50) as shown in Figure 13. If the program does not detect the card's edge (ie., the transition from black to

white) on this line, the scanning process repeats every fiftieth line until the edge is detected. The x-y coordinates of this location are then labeled x(1) and y(1) respectively.

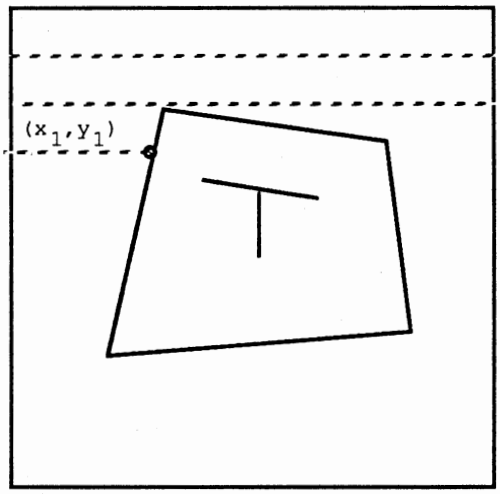


Figure 13. Initial Scan Pattern

Upon detection of an edge, the subroutine ANGLE.FOR is called. This routine uses a second scan algorithm to define a second (x,y) coordinate pair on the same side of the card, as shown in Figure 14. The angle, ANG is calculated as follows

$$ANG = \arctan [(Y_2 - Y_1) / (X_1 - X_2)] \quad (3.1)$$

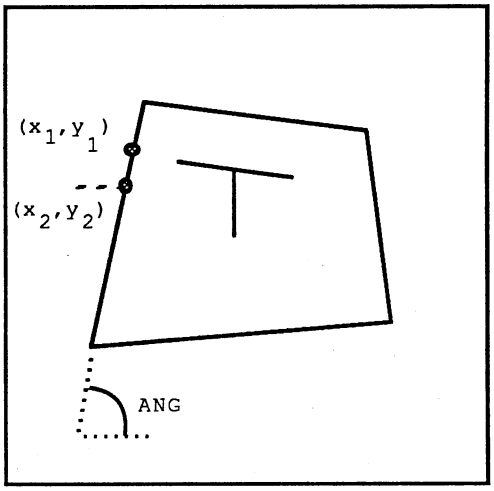


Figure 14. Second Scan Pattern

This angle information is then passed back to the SIDES.FOR subprogram.

The next side to be analyzed is horizontally across the image from the first side, as shown by the numbered scan paths in Figure 15(a). Starting at y = y(1) and x = 512, the image is scanned from right to left until an edge is detected. As for the first side, the program calculates the slope of the new side using the angle algorithm. The next side's calculation starts at y = 512 and x = x(1) and continues scanning from the bottom of the image to the top until the bottom edge is detected. The final side's calculation begins at y = [y(1) + y(4)] / 2 and x = 0 and proceeds from left to right. This procedure works for the card orientation in Figure 15(a); however, there are a total of seven possible orientations, of which this is the

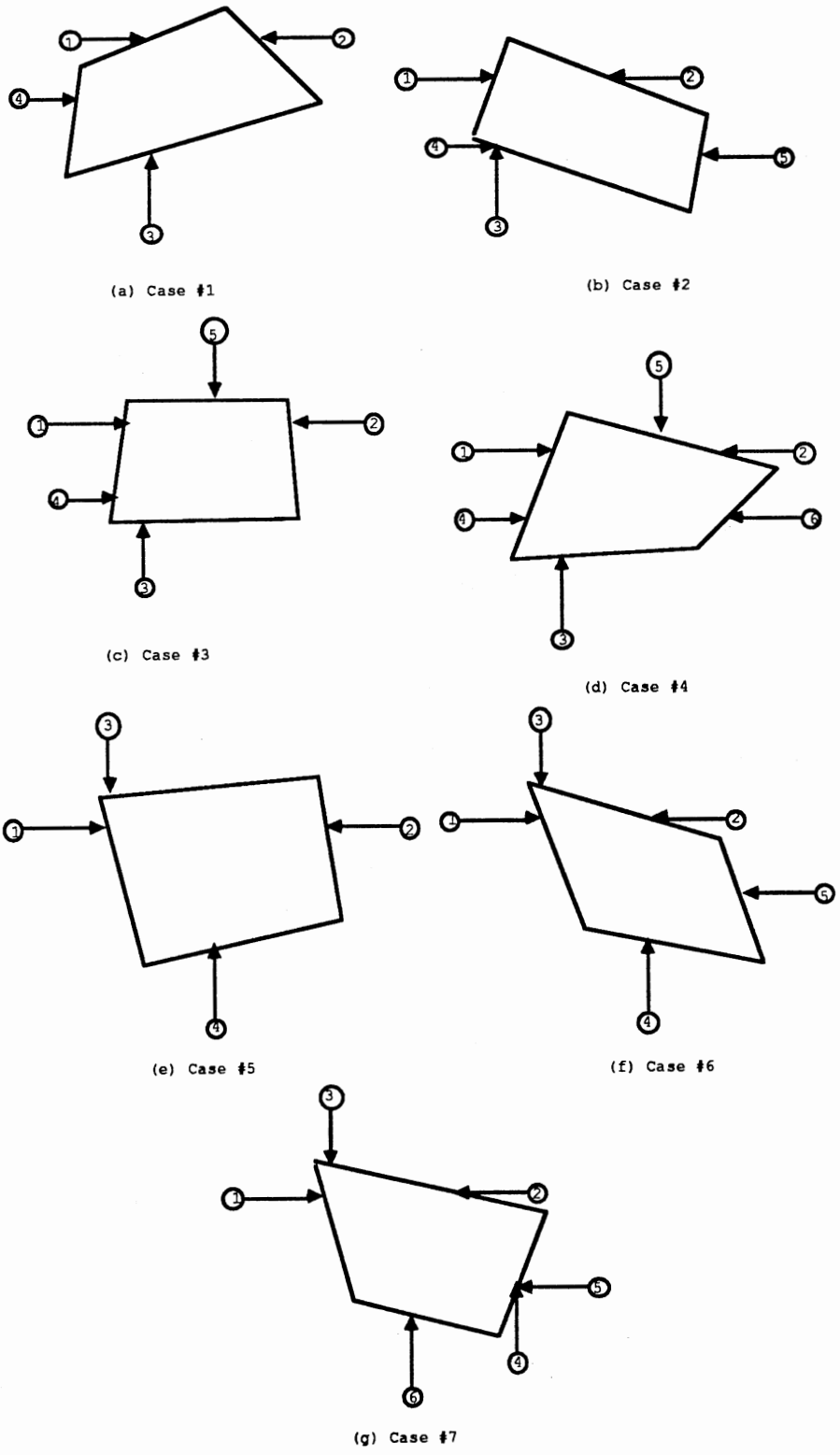


Figure 15. Card Orientations

simplest. The remaining six orientations are shown in Figures 15(b) through 15(g). The first two points are calculated identically for each case, but for the remaining points the tests listed in Table III determine which case should be used for a particular image. Also in Table III are the starting points for the various scan paths in each case. The final task in the side calculations is to number the sides and slopes in a standard orientation. The top-most side is side 1 with its corresponding x(1), y(1), and p(1). The remaining sides and angles are numbered sequentially in a clockwise direction around the card.

Calculate the Card's Corners

The subroutine CORNER.FOR is called to locate the four corners of the card by establishing the relationship between any two adjacent sides. The relationship between sides one and four is shown in an example in Figure 16. Starting with the triangle whose vertices are [X(4),Y(4)], [x_d,Y(4)], and [x_d,Y(1)], the trigonometric relationship

$$\tan (P(4) = [Y(4)-Y(1)] / [X(4) -x_d] \qquad (3.2)$$

is used to solve for x_d, which is given by

$$x_d = X(2) - [Y(4)-Y(1)] / \tan P(4) \qquad (3.3)$$

Then using the triangle formed by the points [X(1),Y(1)],

TABLE III

SCAN PATTERNS AND TESTS

Case Number	Figure Number	Path Number	Starting Position X(I)	Starting Position Y(J)	Scanning Direction	Test
1	15 a	1	1	**	L-R	-
		2	512	Y(1)	R-L	P(1)≥0
		3	X(1)	512	B-T	-
		4	1	[Y(1)+Y(4)]/2	L-R	P(1)≠P(4)
						P(3)≠P(4)
2	15 b	1	1	**	L-R	-
		2	512	Y(1)	R-L	P(1)≥0
		3	X(1)	512	B-T	-
		4	1	[Y(1)+Y(3)]/2	L-R	P(3)=P(4)
		5	512	Y(3)	R-L	-
3	15 c	1	1	**	L-R	-
		2	512	Y(1)	R-L	P(1)≥0
		3	X(1)	512	B-T	-
		4	1	[Y(1)+Y(3)]/2	L-R	P(1)=P(4)
		5	[x(1)+x(2)]/2	1	T-B	P(3)≠P(2)
4	15 d	1	1	**	L-R	-
		2	512	Y(1)	R-L	P(1)≥0
		3	X(1)	512	B-T	-
		4	1	[Y(1)+Y(3)]/2	L-R	P(1)=P(4)
		5	[x(1)+x(2)]/2	1	T-B	P(3)=P(2)
		6	512	[Y(2)+Y(1)]/2	R-L	-
5	15 e	1	1	**	L-R	-
		2	512	Y(1)	R-L	P(1)<0
		3	X(1)	512	T-B	-
		4	512	[Y(1)+Y(2)]/2	B-T	P(3)≠P(2)
6	15 f	1	1	**	L-R	-
		2	512	Y(1)	R-L	P(1)<0
		3	X(1)	1	T-B	-
		4	X(2)	512	B-T	P(3)=P(2)
		5	512	[Y(2)+Y(3)]/2	R-L	P(3)≠P(4)
7	15 g	1	1	**	L-R	-
		2	512	Y(1)	R-L	P(1)<0
		3	X(4)	1	T-B	-
		4	X(1)	512	B-T	P(3)=P(2)
		5	512	Y(4)	R-L	P(3)=P(4)
		6	[X(4)+X(2)]/2	512	B-T	-

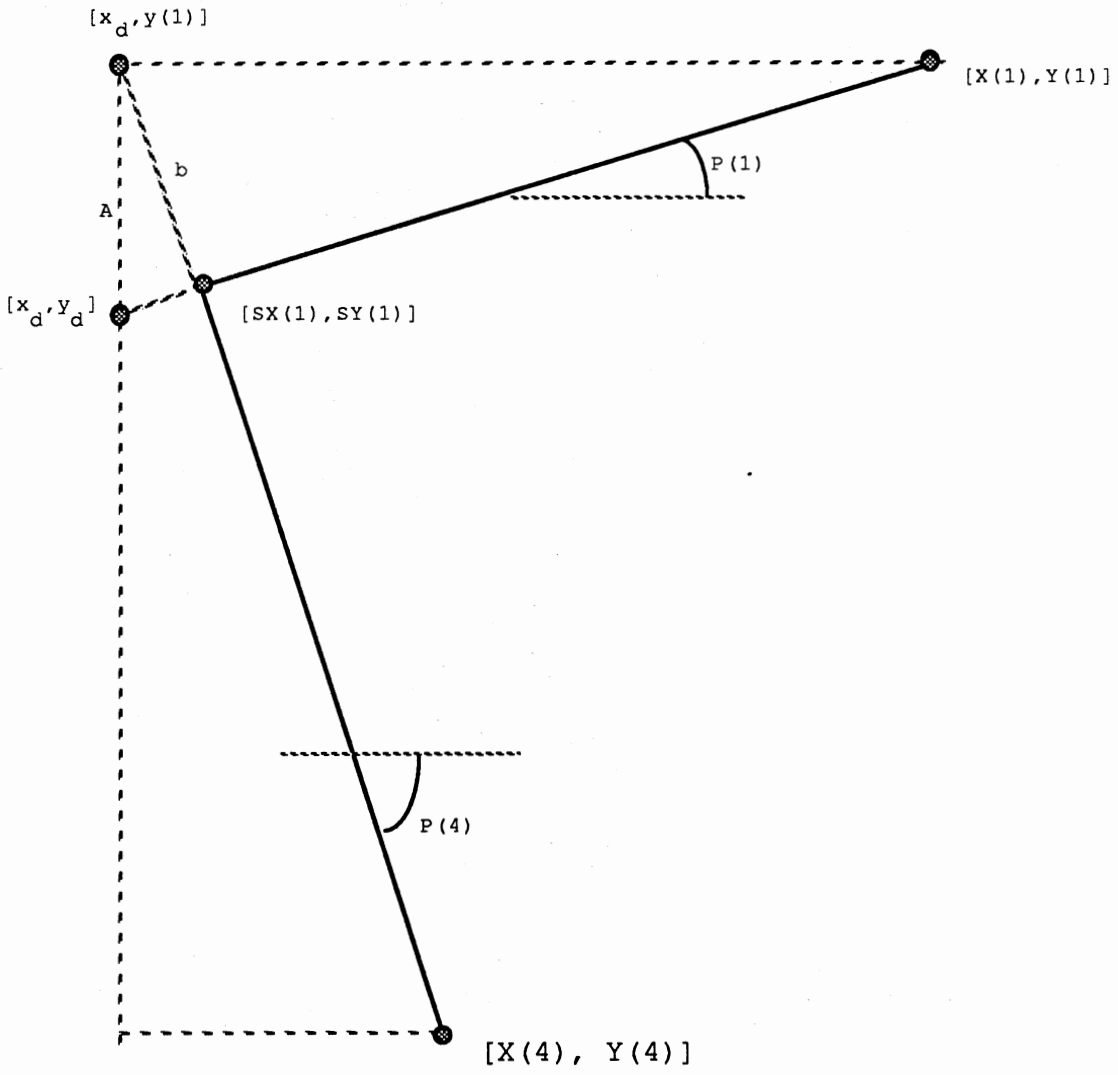


Figure 16. Corner Calculation

$[x_d, Y(1)]$, and $[x_d, y_d]$, the coordinate y_d is given by

$$y_d = Y(3) - \tan P(1) * [X(1) - X(3)] \tag{3.4}$$

The next step is to use the law of sines

$$\frac{yd - Y(1)}{\sin [P(1) + P(4)]} = \frac{b}{\sin [\pi/2 - P(1)]} \tag{3.5}$$

to solve for the length b in Figure 16

$$b = [y_d - Y(1)] * [\sin (\pi/2 - P(1))] / [\sin P(1) + \sin P(4)] \tag{3.6}$$

The last step is to calculate the actual x and y coordinates of the corner point, as given by

$$CX(1) = x_d + b * \cos P(4) \tag{3.7}$$

$$CY(1) = y_d + b * \sin P(4) \tag{3.8}$$

The remaining corners of the card are calculated using similar techniques.

Update Angle and Corner Calculations

The initial angle calculations for each case are inaccurate due to the small Δx and Δy used to find the second card edge intersection points. Small Δx and Δy values are used in an attempt to guarantee that the second edge point is on the same side as the initial point. To increase the accuracy of the angle calculations, larger Δx and Δy values are needed. Using the previously calculated corner coordinates, new algorithms scan the card image to find new

edge intersection points near the corners. These new x and y side coordinates are used to update the side angle calculations and the corner coordinate calculations by calling the subroutine CORNER.FOR again.

Define the Character Area

The area of the card or label that immediately surrounds the character is defined as the character area and is established in the subroutine CHARACTER.FOR. If there are multiple characters per card, the character string can be separated into consecutive single-character areas prior to image processing. It is assumed in the single character case that the character is centered on the card. Thus, the center of the card corresponds to the center of the character area in this case. To calculate the card's center point, the program first calculates the slopes of the card's two diagonals, as shown in Figure 17. The slopes are given by

$$M(1) = \arctan \{[CY(4) - CY(2)]/[CX(2) - CX(4)]\} \quad (3.9)$$

$$M(2) = \arctan \{[CY(1) - CY(3)]/[CX(3) - CX(1)]\} \quad (3.10)$$

The intersection of the two diagonals is the center of the card, and therefore the center of the character area. To calculate the center point, the program uses the equations above to establish the following relationships

$$\tan M(1) = [\, CY(4) - CCY \,] \,/\, [\, CCX - CX(4) \,] \quad (3.11)$$

$$\tan M(2) = [\, CY(1) - CCY \,] \,/\, [\, CCX - CX(1) \,] \quad (3.12)$$

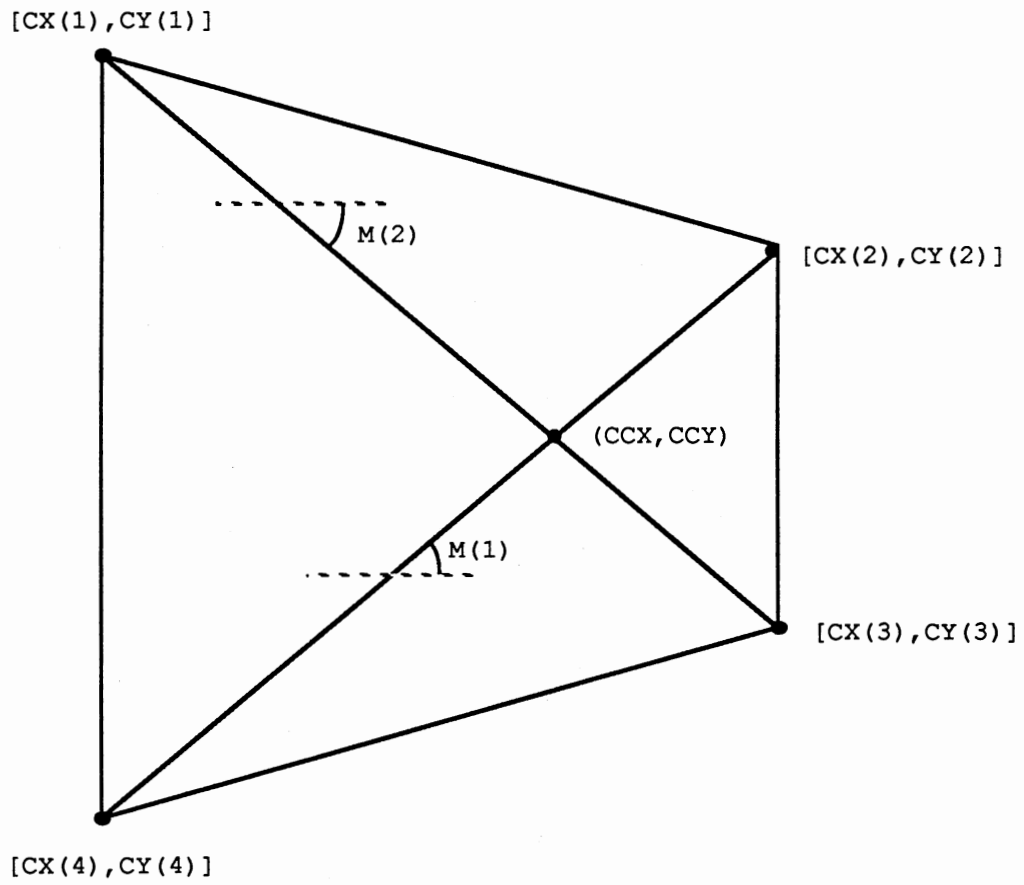


Figure 17. Center Point Calculation

Solving for equation (3.11) for CCY gives

$$CCY = CY(4) - \tan M(1) * [\ CCX - CX(4)\ ] \qquad (3.13)$$

Substituting this into equation (3.12) and solving for CCX gives

$$CCX = \frac{CY(1)-CY(4)-CX(4)*\tan M(1)+CX(1)*\tan M(2)}{[\tan M(2) - \tan M(1)]} \qquad (3.14)$$

Once CCX has been solved for, the value for CCY is obtained

using the calculated value of CCX.

The final step in defining the character area is to identify its corner coordinates. In order to calculate the corner coordinates, the ideal card and the ideal character lengths must first be defined. Figure 18 shows these relationships, where L is the ideal card length and l is the ideal character length. Since the entire field of view of the COMTAL system is 512 x 512 pixels, then by letting L = 512, the largest possible image is obtained. Similarly, the value of the ideal character length is set to an arbitrary value of l=128 to completely surround the character.

Since the image is skewed, the lengths a and b in Figure 18 depend upon the degree of tilt away from the camera. Since the lengths from each corner to the center can be calculated, the lengths from the corner of the card to the corner of the character area can be obtained using the equation

$$CA(i) = AX(i) - c * AX(i) / L \qquad (3.16)$$

These lengths are then used to calculate the corner coordinates of the character area. For example, corner three is given by

$$SX(3) = CX(3) - CA(3) * \cos M(2) \qquad (3.17)$$

$$SY(3) = CY(3) - CA(3) * \sin M(2) \qquad (3.18)$$

The remaining three corners are calculated using similar equations.

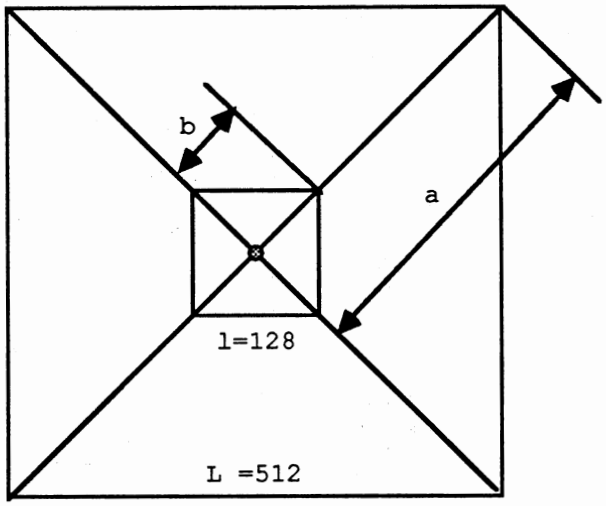


Figure 18. Ideal Image

Threshold the Character Area

Once the card parameters have been calculated, the original image is thresholded again using the subroutine THRESHOLD.FOR. While the first thresholding helped define the edges of the card, the second threshold filter uses a different threshold value to better clarify the character image itself prior to the reconstruction process. The threshold routine first prompts the operator for the new threshold value. The remainder of the routine is a looping function which performs the thresholding. The resulting

image is passed back to the executive routine for more processing.

Rotate the Image

If necessary, the image is rotated, using the subroutine ROTATE.FOR, such that one side of the character area is aligned along the horizontal, which allows for easier expansion in the x direction. The rotation process begins by establishing vertical and horizontal quadrant lines through the character area, as show in Figure 19. These quadrant lines simulate the lines that would be seen if the original card were divided into four square quadrants before the image was captured. M(3) and M(4) denote the angles between the quadrant lines and the horizontal axis. These angles are given by

$$M(3) = [\ P(1) + P(3)\ ] / 2 \qquad (3.19)$$

$$M(4) = [\ P(2) + P(4)\ ] / 2 \qquad (3.20)$$

where P(i) is the angle for side i, as calculated in the SIDES.FOR and ANGLE.FOR subroutines.

At this point, the image data needs to be rotated by some angle of rotation θ. By letting θ = M(4) and using the rotation coordinate transform described in Chapter II, the image is rotated until the horizontal quadrant line is aligned on the horizontal axis, that is, until M(4) = 0. In this way, the character area data is mapped from its original

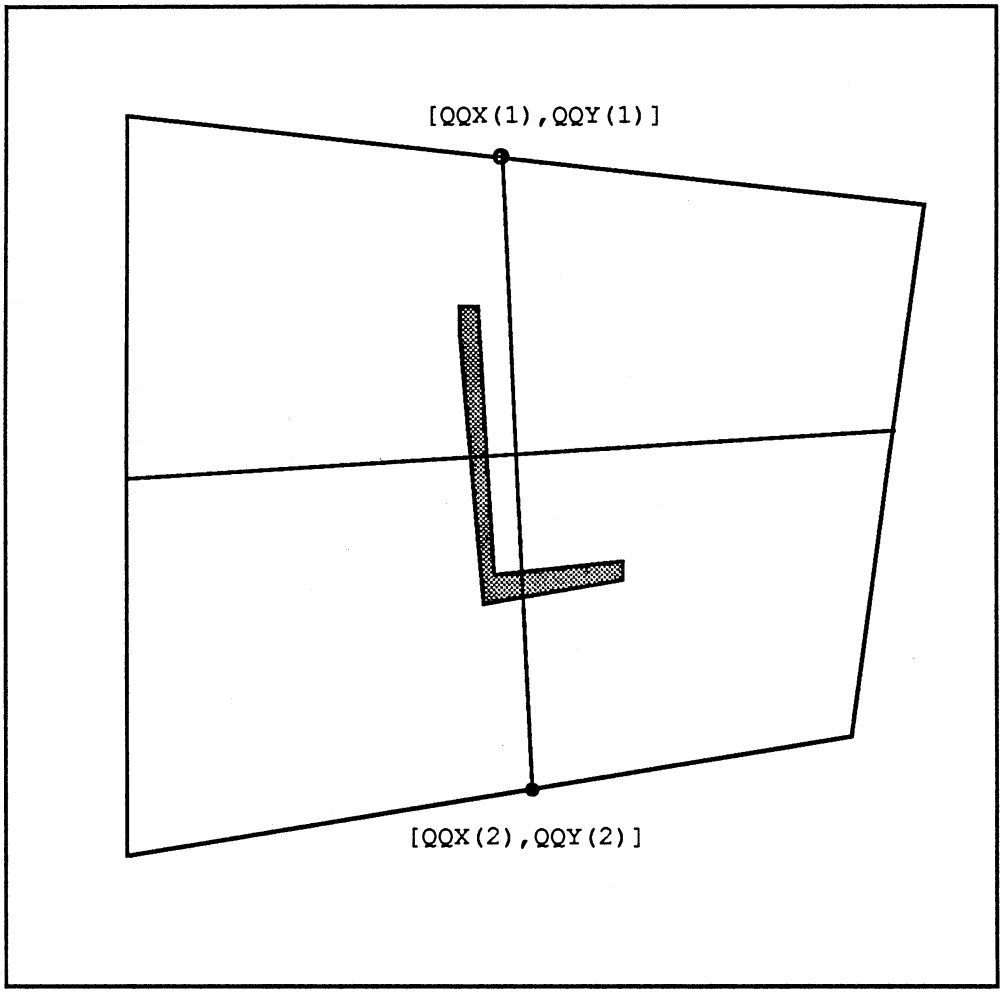


Figure 19. Quadrant Line

coordinate system into a standard coordinate system, thus standardizing the image for expansion.

All of the character area parameters must be transformed to the new coordinate basis by applying the rotational transforms. The new corner coordinates are given by

$$SX(i) = SX(i) * \cos M(4) + SY(i) * \sin M(4) \qquad (3.21)$$

$$SY(i) = SY(i) * \cos M(4) - SX(i) * \cos M(4) \qquad (3.22)$$

The angles of the sides are also transformed to give

$$P(i) = P(i) - M(4) \tag{3.23}$$

The last parameter calculated before the expansion are the points [QQX(1),QQY(1)] and [QQX(2),QQY(2)], which are the intersection points of the vertical quadrant line and the top and bottom sides of the character area. These coordinates are defined by

$$QQX(1) = \frac{CCY - SY(1) + SX(1) * \tan P(1) + CCX * \tan P(3)}{\tan P(3) + \tan P(1)} \tag{3.24}$$

$$QQX(2) = \frac{SY(4) - CCY + SX(4) * \tan P(3) - CCX * \tan M(3)}{\tan P(3) + \tan P(1)} \tag{3.25}$$

$$QQY(1) = CCY - \tan P(3) * (QQX(1) - CCX) \tag{3.26}$$

$$QQY(2) = CCY + \tan M(3) * (CCX - QQX(2)) \tag{3.27}$$

Determine Character Area Shape

When an image is captured, the card can be at any of an infinite number of orientations relative to the camera, resulting in a captured image which is skewed. These images fall into one of twelve shape categories. Thus, a reliable testing procedure is required to determine the appropriate shape category for a particular image. This testing procedure, performed in the subroutine AREA.FOR, uses a tree-branching structure as shown in Figure 20. The test structure starts at the top of the figure and the blocks at

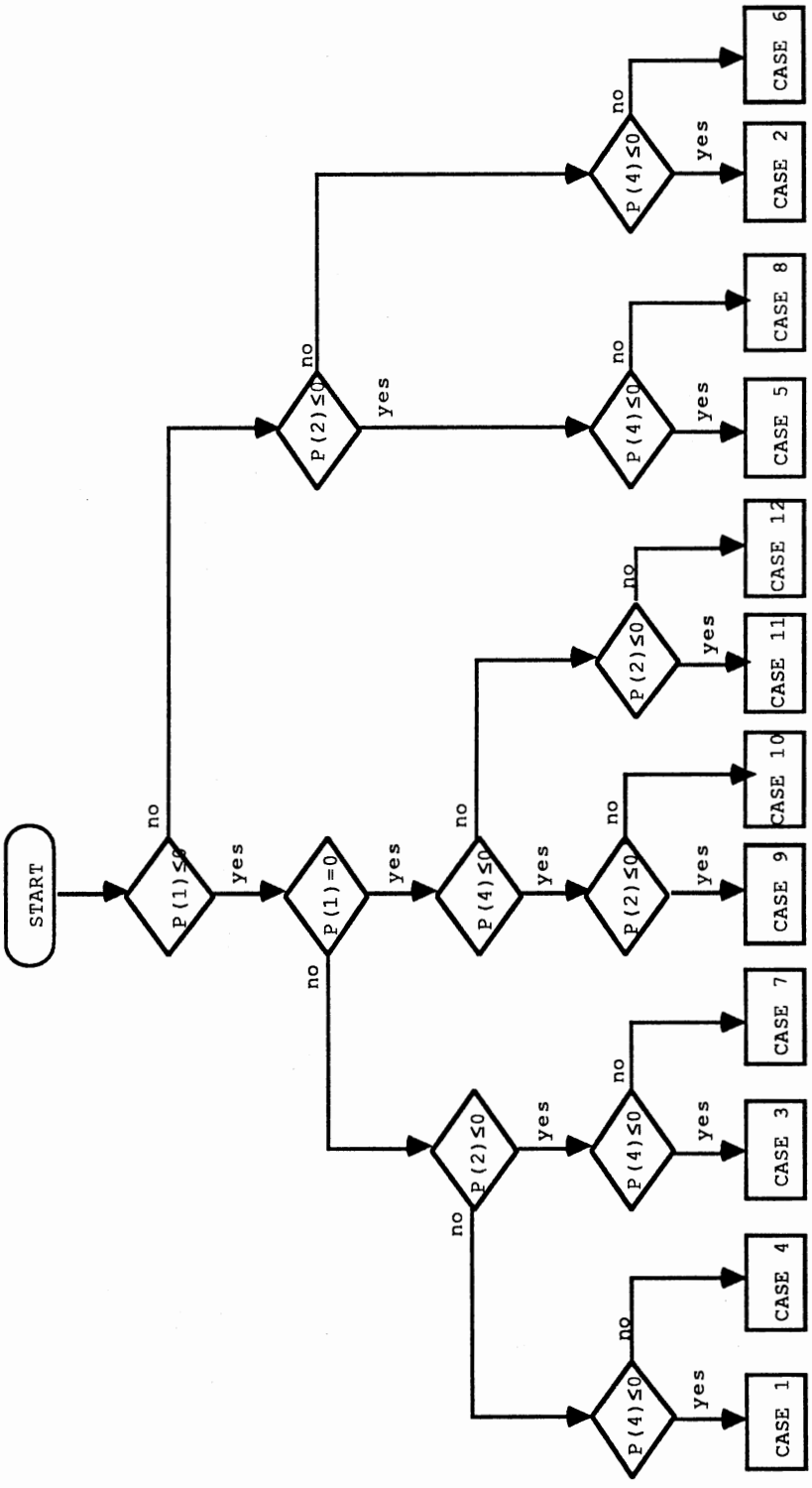


Figure 20. Area Decision Tree

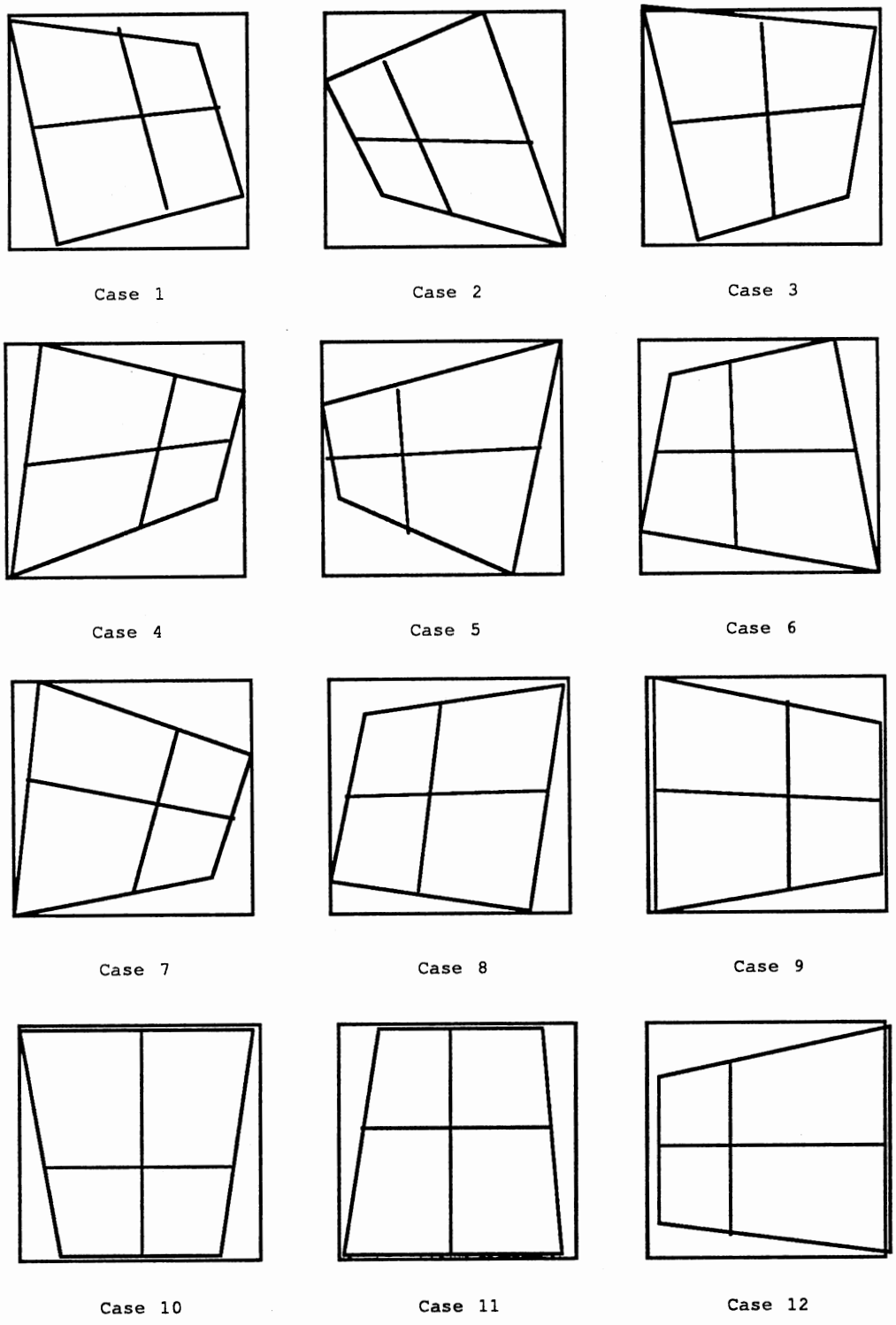


Figure 21. Area Configurations

the bottom of the figure correspond to the twelve different categories. Figure 21 shows examples of the different configurations.

Once the shape case of the image is determined, the program calculates a ratio for both the left and right quadrants of the character area. These ratios, RA(i) and RB(i), represent the amount of expansion required to completely reconstruct the image in the x direction. The program calculates the array of the lengths RA(i) and RB(i) beginning on the top line and proceeding down to the bottom. The lengths are calculated using trigonometric relationships. For example, for a category I image with SY(2) < i < SY(3), the lengths RA(i) and RB(i) are given by

$$RA(i) = QX(i) - LS * \tan [\pi/2 - P(4)] \quad (3.28)$$

$$RB(i) = RS - QX(i) - CY(3) - i * \tan [\pi/2 - P(2)] \quad (3.29)$$

Expand Image in X Direction

The subroutine XEXPAND.FOR uses the values of RA(i) and RB(i) obtained in the subroutine AREA.FOR to calculate the required ratio of expansion for a given line, i. The ratio is the magnification coefficient for each line; it stretches each line of the skewed character to its correct length. The ratios are given by

$$EXPA = [\ \text{Ideal Length}\ ] / RA(i) \quad (3.30)$$

$$EXPB = [\ \text{Ideal Length}\ ] / RB(i) \quad (3.31)$$

where A and B refer to the left and right sides of the character area, respectively, and the Ideal Length of each side is 64 pixels.

Before the expansion can be performed, an accurate representation of the image is obtained using a run-length encoding technique. This technique scans the image from left to right to count the number of consecutive black or white pixels using a run-length counter, COUNT. When a transition from a white pixel to a black pixel (or vice versa) is encountered, the transition counter, ATRAN, is incremented and the number of pixels in the run length is stored in the run-length array, ACODE, as given by

$$\text{ACODE(ATRAN)} = \text{COUNT} \tag{3.32}$$

Similarly, the run-length array for the right side of the image is given by

$$\text{BCODE(BTRAN)} = \text{COUNT} \tag{3.33}$$

where BTRAN and BCODE are the transition counter and the run-length array for the right side. The running count of the total number of pixels is updated using the relationship

$$\text{ATOTAL} = \text{ATOTAL} + \text{COUNT} \tag{3.34}$$

$$\text{BTOTAL} = \text{BTOTAL} + \text{COUNT} \tag{3.35}$$

These counters are used by the filling algorithms described below.

Upon completion of the run-length encoding process for a particular line, the data filling algorithm determines the expanded number of black and white pixels to fill into the expanded image matrix. The expanded data is calculated by multiplying the pixel count of each black and white run length by the expansion ratio for that line, as given by

$$AFILL(i) = ACODE(i) * EXPA \tag{3.36}$$

$$BFILL(i) = BCODE(i) * EXPB \tag{3.37}$$

where i and j are loop counters that vary from one to ATRAN and BTRAN respectively. AFILL and BFILL represent the expanded number of consecutive black and white pixels to be filled into the new image matrix.

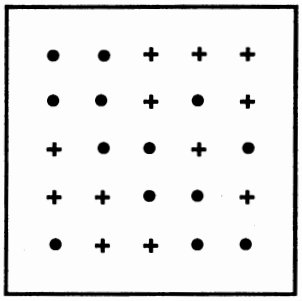
An example of the x-expansion process is shown in Figure 22(a), which represents an original image and Figure 22(b), which represents the x-expanded image using a magnification ratio of 2:1. Examining the third row of the original image, for example, the number of transitions is four, and the run length encoded data is given by

$$CODE(1) = 1 \tag{3.38}$$

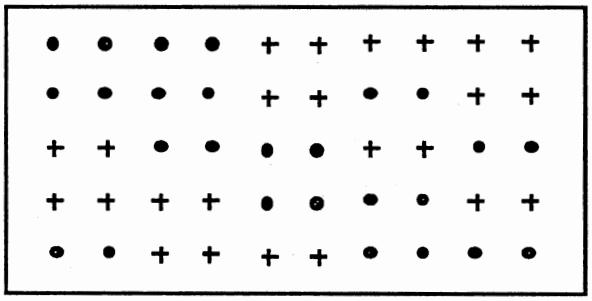
$$CODE(2) = 2 \tag{3.39}$$

$$CODE(3) = 2 \tag{3.40}$$

$$CODE(4) = 2 \tag{3.41}$$



(a) Original Image



(b) Expanded Image

Figure 22. Magnification Example

Using the 2:1 magnification ratio gives

$$FILL(1) = 1 \tag{3.42}$$

$$FILL(2) = 2 \tag{3.43}$$

$$FILL(3) = 2 \tag{3.44}$$

$$FILL(4) = 2 \tag{3.45}$$

Filling in the new image matrix with the values above result in the expanded image shown in Figure 22(b).

One problem with the filling routine that is created by the run-length counting routine is that the counting is performed over a rectangular area which is actually larger than the skewed character area. The result is that more white are, or zeros, per line are counted than need to be expanded. If the filling routine filled in all of the expanded data into the new image matrix from left to right, the extra zeros that had been expanded would distort the results. For this reason, the filling routine first calculates the total expanded length of each side (ATOTAL * EXPA and BTOTAL * EXPB). If these expanded lengths are greater than the ideal length, 64 pixels, then the extraneous data on the left and right edges of the character area are truncated. The filling routine thus stores only the data expanded from the skewed character area in the new image matrix. The resulting x-expanded image is in one of the forms shown in Figure 23.

Expand Image in Y Direction

The subroutine YEXPAND.FOR is called by the executive program to complete the reconstruction by expanding the image vertically. The y-expansion process is identical to the x-expansion process. First, the total length of the black and white run-length counts are calculated for each pixel column from the horizontal quadrant line to the top and to 24. The y-expansion ratio for each column is calculated by

comparing the ideal image length to the length of the character area column. The white and black counted lengths are then expanded vertically, working from the horizontal quadrant line up for the top of the image and down for the bottom portion. Again, any extraneous data outside of the 128 pixel height is truncated so that the new image contains only data expanded from the character area. The reconstructed character is now ready for classification.

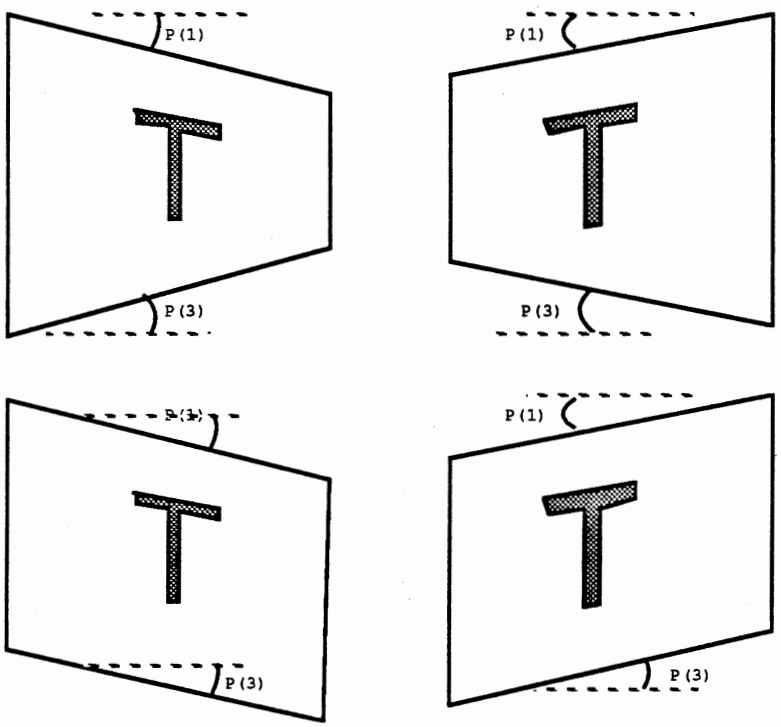


Figure 23. X Expanded Image Types

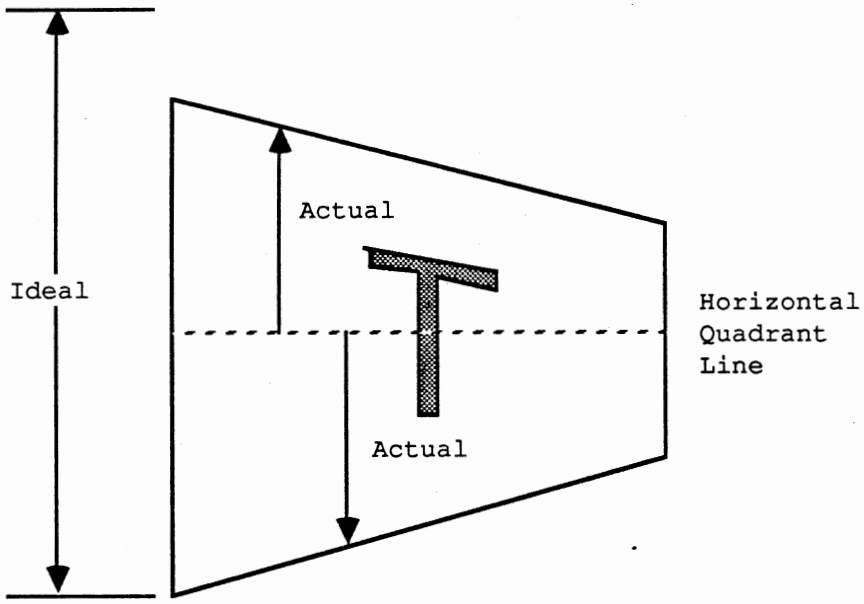


Figure 24 Y Expansion Calculation

Segment the Image

The classification and interpretation phases begin by calling the subroutine OFSET.FOR, which scans the character area of the reconstructed image to find the outer boundaries of the character as shown in Figure 25. Since a pixel by pixel comparison algorithm is used to recognize the image, the placement of the reconstructed character within the character area window is critical. Since the true center of the card is a fixed point, the character can be moved to this

point using a simple translation. The centroid of the character is given by

$$CNX = (RS - LS) / 2 \quad (3.46)$$

$$CNY = (BT - TP) / 2 \quad (3.47)$$

Knowing that the true center point of the 128 x 128 pixel area is the point (64,64), the offset calculations for the translation are given by

$$OFSTX = TRUEX - CNX \quad (3.48)$$

$$OFSTY = TRUEY - CNY \quad (3.49)$$

In this way, the character image is translated to the center of the character area window and can be directly compared to the stored character templates.

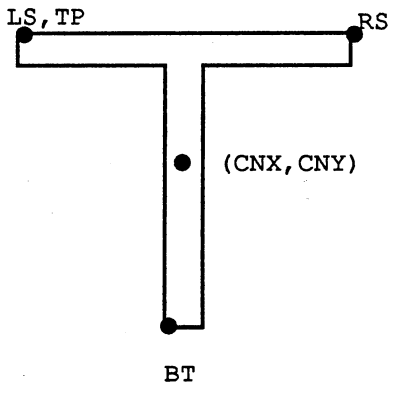


Figure 25. Translate Calculation

Recognize the Character

A template matching approach is used to recognize the reconstructed character. To perform the comparison, each pixel in the reconstructed image is compared to the corresponding pixel in each template. If both points are a black pixel, a counter is incremented. In the same software loop, another counter is incremented for every black pixel in the template. The efficiency of the pixel matching is given by

$$\text{efficiency} = \frac{\Sigma \text{ black points in agreement}}{\Sigma \text{ black points in template}} \tag{3.50}$$

For a particular reconstructed character, the template with the highest matching efficiency is the character recognized by the system. The efficiency results for each template are printed by the subroutine OUTPUT.FOR.

CHAPTER IV

DATA GATHERING AND RESULTS

This chapter describes how the COMTAL system is used to capture image data and how the threshold values and camera focal lengths were determined. It then presents an analysis of the data gathered for this research.

Data Gathering Process

The data gathering process begins with physically setting up the COMTAL camera system to capture an image. The card is set on a black cloth background beneath the camera lens. The lighting must be adjusted to illuminate the card while minimizing light leakage around the edges of the camera's field of view. Captured images that have a border of light at the edges must be discarded, the lighting adjusted and the image recaptured. Once the lighting is established, the height of the camera with respect to the image plane is adjusted to ensure that proper magnification ratios are used in the expansion algorithms. The angles of rotation in the xz and yz planes, as shown in Figure 26, are measured and noted for subsequent analysis. The last steps in the image setup are the focus and aperture adjustments on the camera to produce the best possible captured image.

The next portion of the data gathering process is to capture and store the image. This is accomplished by using several COMTAL commands to freeze the displayed image into an image plane on the COMTAL. Once the image is in a plane, a VAX transfer command stores it into a VAX image data file for processing by the programs as described in Chapter III.

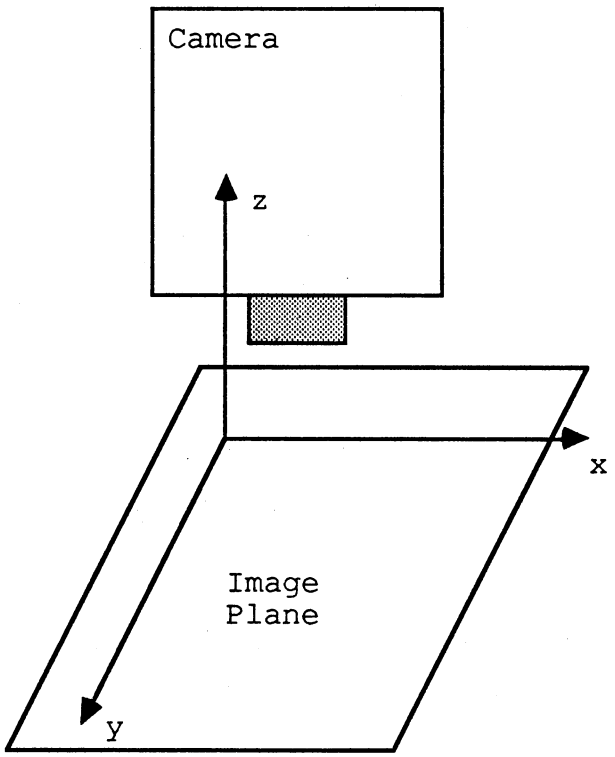


Figure 26. Image Plane

Several problems can arise during the image setup and capture processes due to the system's sensitivity to lighting, focus, and aperture adjustments. If the lighting is too close to the card, reflections from the background material add unwanted noise to the captured images. On the other hand, poor lighting can produce dark images that are difficult to analyze. The combination of the three factors listed above contributes to the difficulty in achieving reproducible image recognition results.

After an image is captured and stored, the executive routine begins the image processing. Near the completion of the executive's operation, the results of the template matching algorithms, including the template for the final recognized character, are displayed to the operator's terminal. Also, the 512 x 512 display image file is transferred to the COMTAL for a visual confirmation of a successful image reconstruction.

Threshold and Focal Length Measurements

As was previously discussed, two threshold values are used in the algorithms. The first threshold value is used to sharply define the edge of the card against the black background. This threshold was determined by trial and error using visual inspection of the threshold image versus the original image. The best visual results were attained using threshold values below 100.

The second threshold value is more important for

character recognition because it differentiates the character from the white card. This threshold setting is critical due to the system's sensitivity to the focus, lighting, and aperture settings. The settings create varying gray levels around the edges of the character image. As the threshold value is increased, additional black pixels are added to the character, while if the threshold is decreased, a significant portion of the character's edges may disappear. The optimum character threshold value for this research was determined by testing three characters at various threshold values. Each character was captured as a non-rotated flat image and then processed using the expansion algorithms at the different thresholds. The resulting efficiencies of the expand image compared to each template are given in Table IV of Appendix B and in graphical form in Figures 27, 28, and 29.

In general, the efficiency increases as the threshold value increases, until reaching a break point where the efficiency decreases sharply. Even though the results show that the peak efficiency is at a threshold value between 135 and 140, the reconstructed character images look better at a slightly lower threshold. At the higher threshold values, too many additional border gray-level pixels are added to the character, distorting its proportions. Thus, the preliminary conclusion is that the optimal threshold is between 100 and 125.

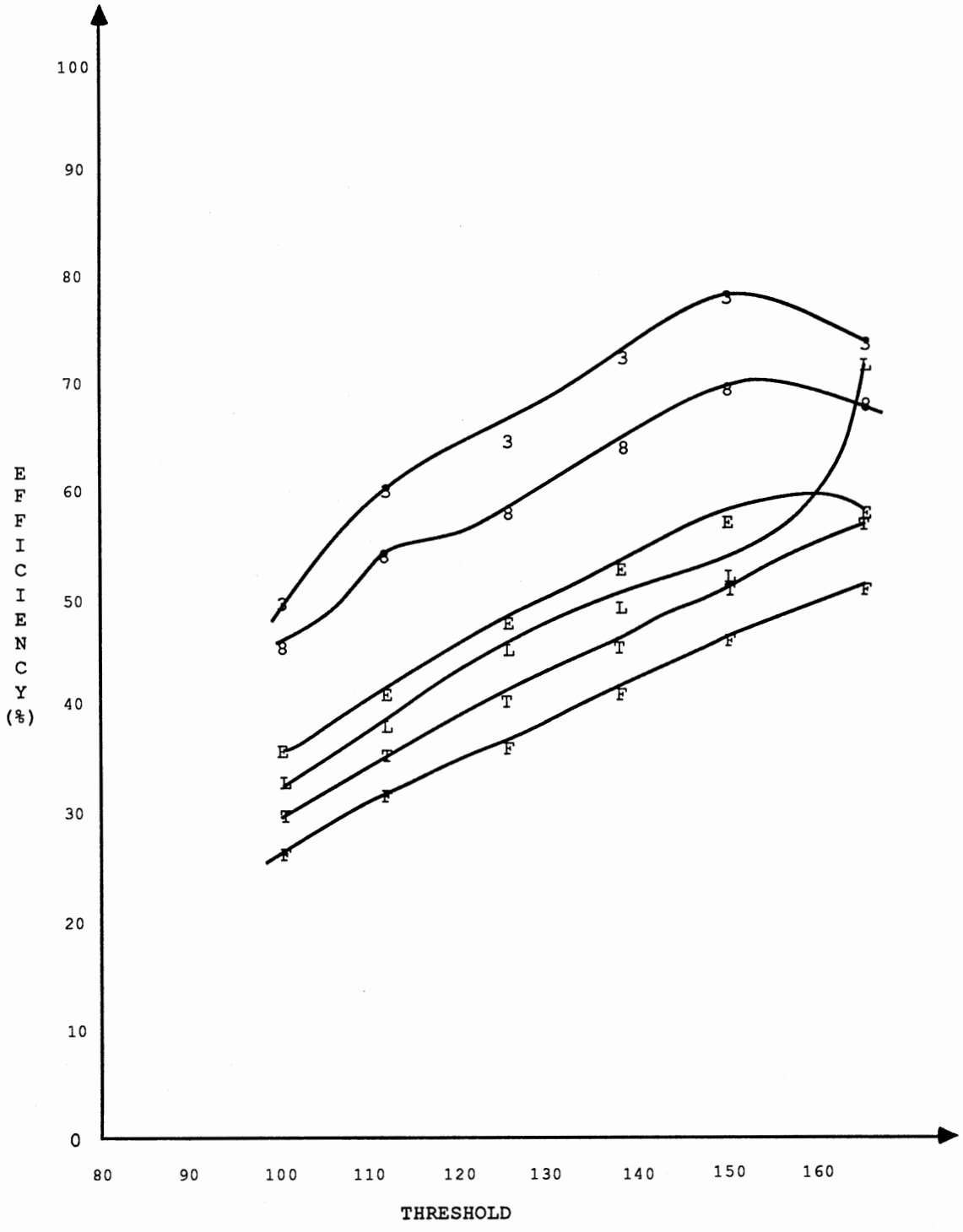


Figure 27. Threshold Test Results (3)

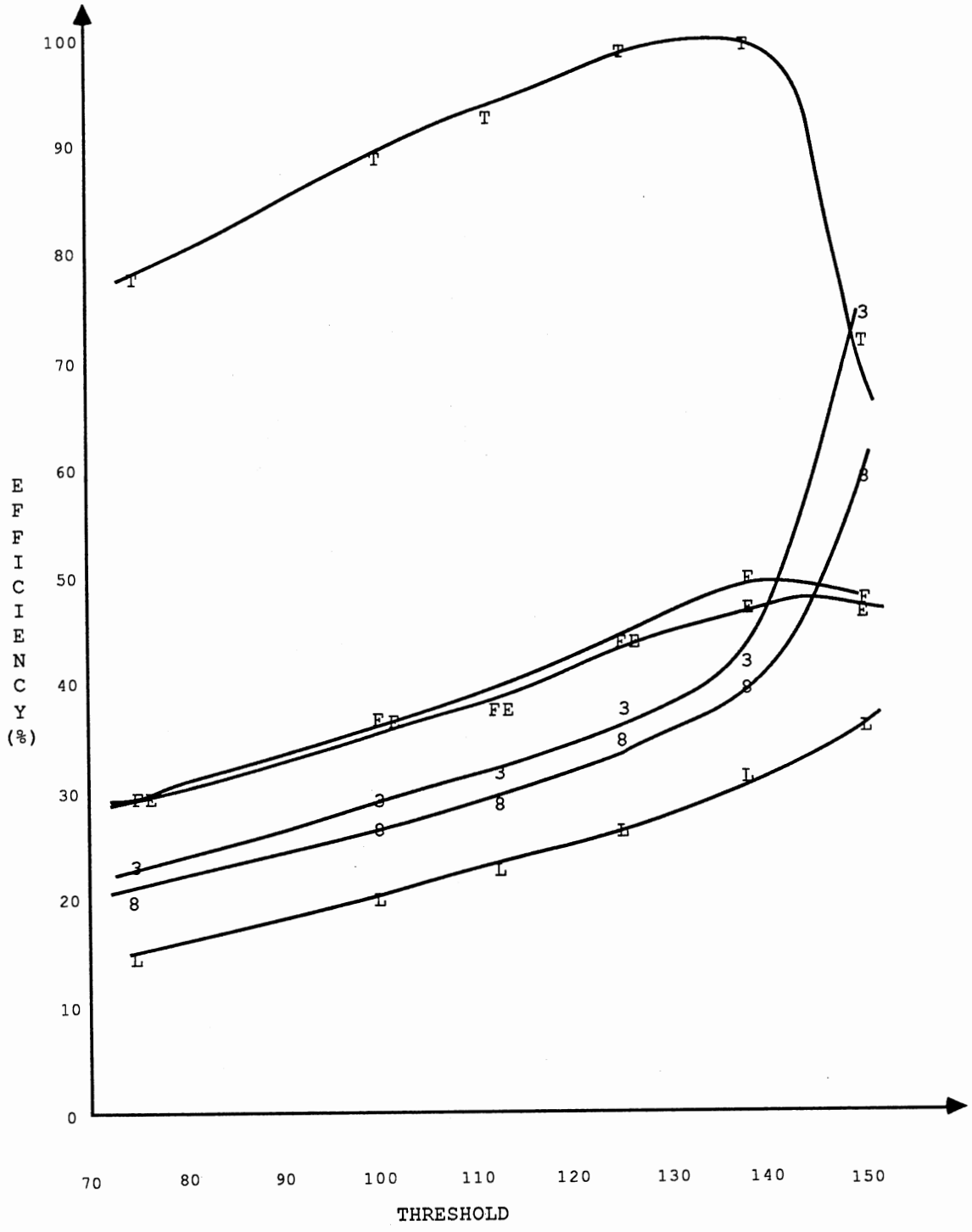


Figure 28. Threshold Test Results (T)

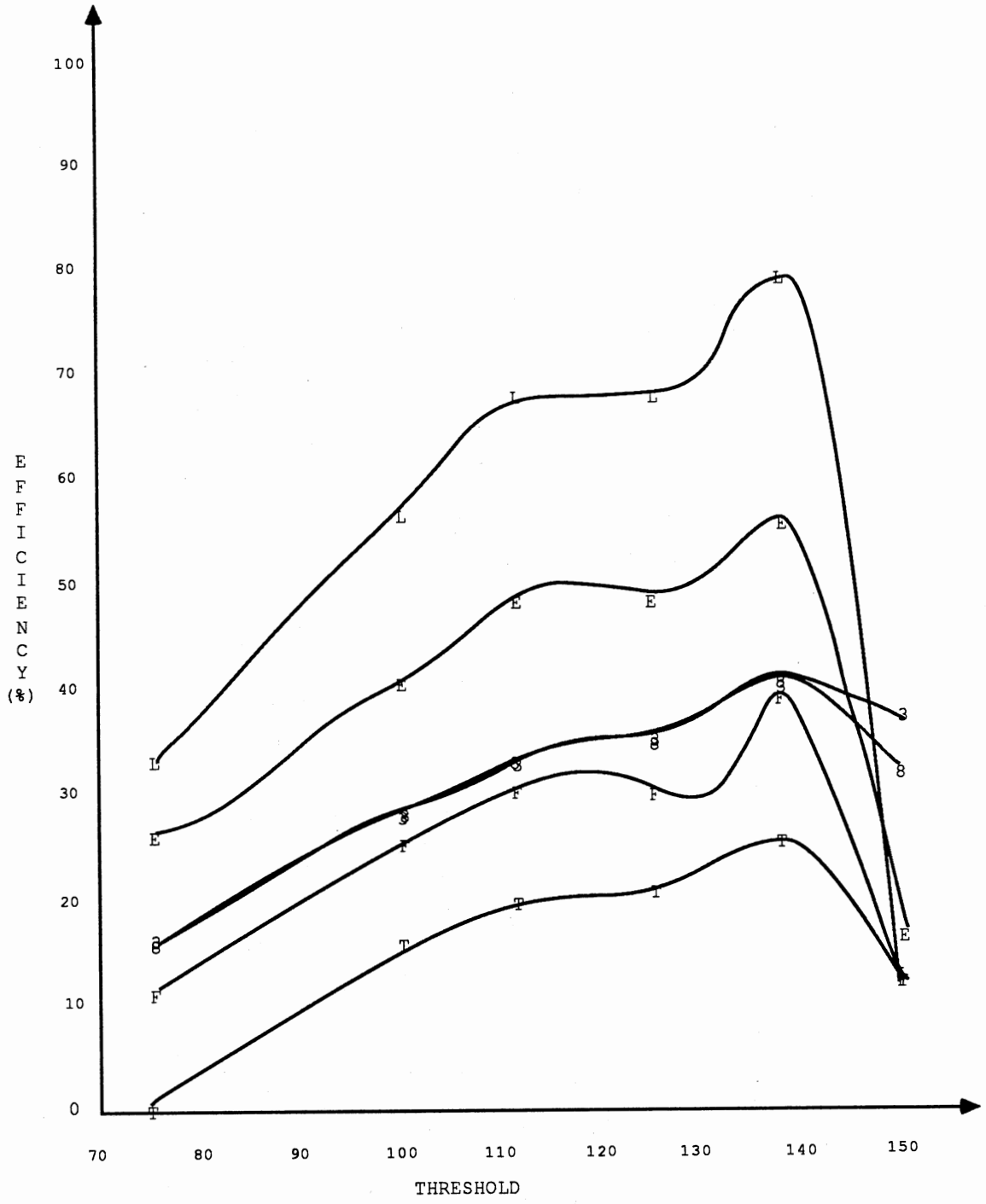


Figure 29. Threshold Test Results (L)

The focal length measurement determines the optimal distance between the card and the camera lens for the best character recognition. The focal length test measurements were taken using a non-rotated flat character "8" at three threshold settings; 100, 112, and 125. The focal length varied from 13.5 inches to 19.0 inches when the images were captured. The resulting reconstruction efficiencies for the "8" versus each template appear in Table V and Figures 30, 31, and 32.

In two of the figures the highest efficiencies are at a focal length of 16.5 inches. Also, note that in Figure 31, where the character threshold value is 112, the recognition efficiency exhibits a very flat response to the changing focal length. These two facts plus the results of the previous threshold measurements indicate that a threshold value of 115 and a focal length of 16.5 inches are the optimal conditions for image capturing. These values are used for the remainder of the data collection.

Single Axis Rotation Results

The goal of the single axis of rotation measurements is to establish performance boundaries for the character recognition process. Several characters were used in four separate cases defined by positive and negative z rotation along the x axis and along the y axis. Letting Ψ be the rotation about the x axis and Θ the rotation about the y axis, Figures 33, 34, 35, and 36 shows the reconstruction

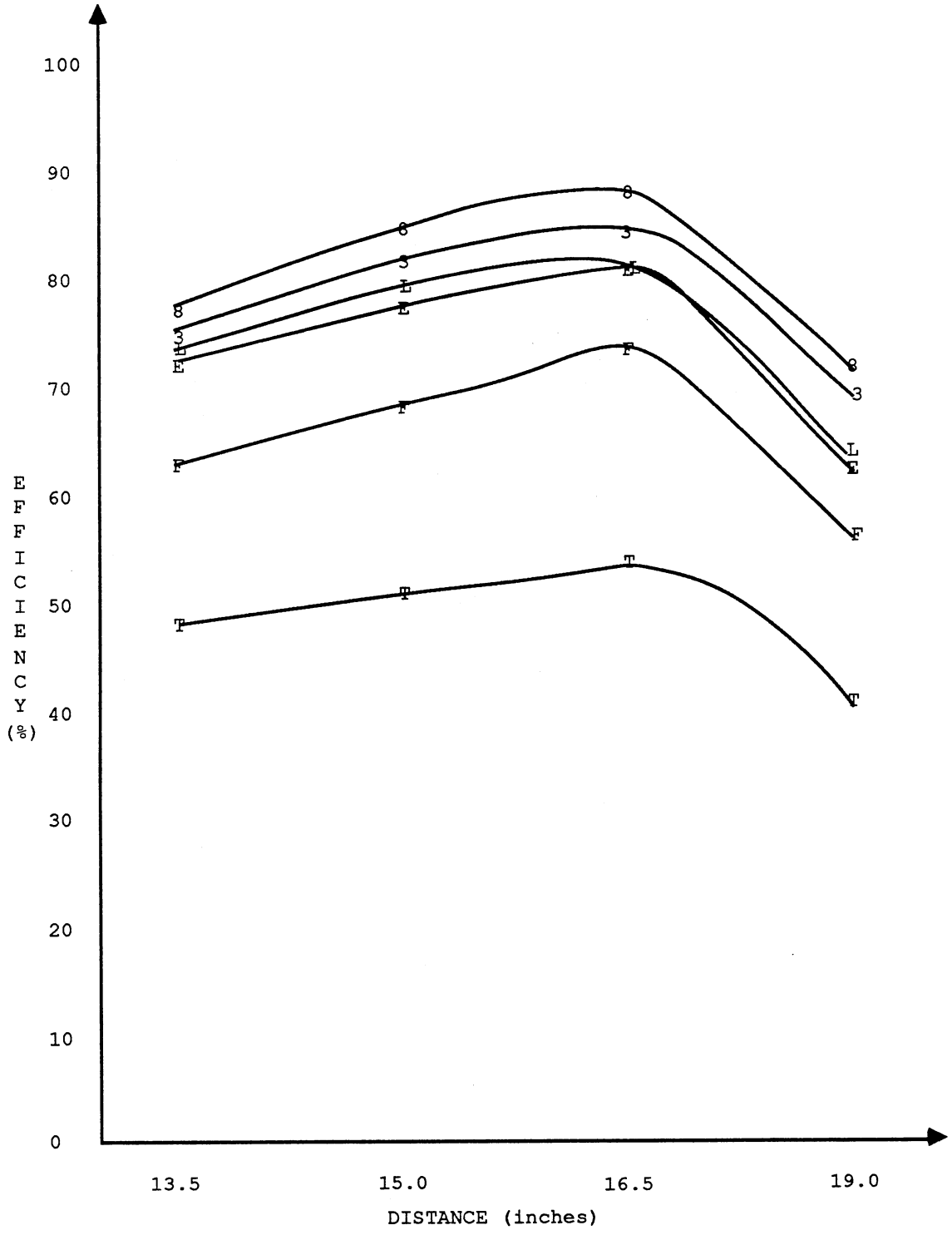


Figure 30. Focal Length Test Results (THR = 100)

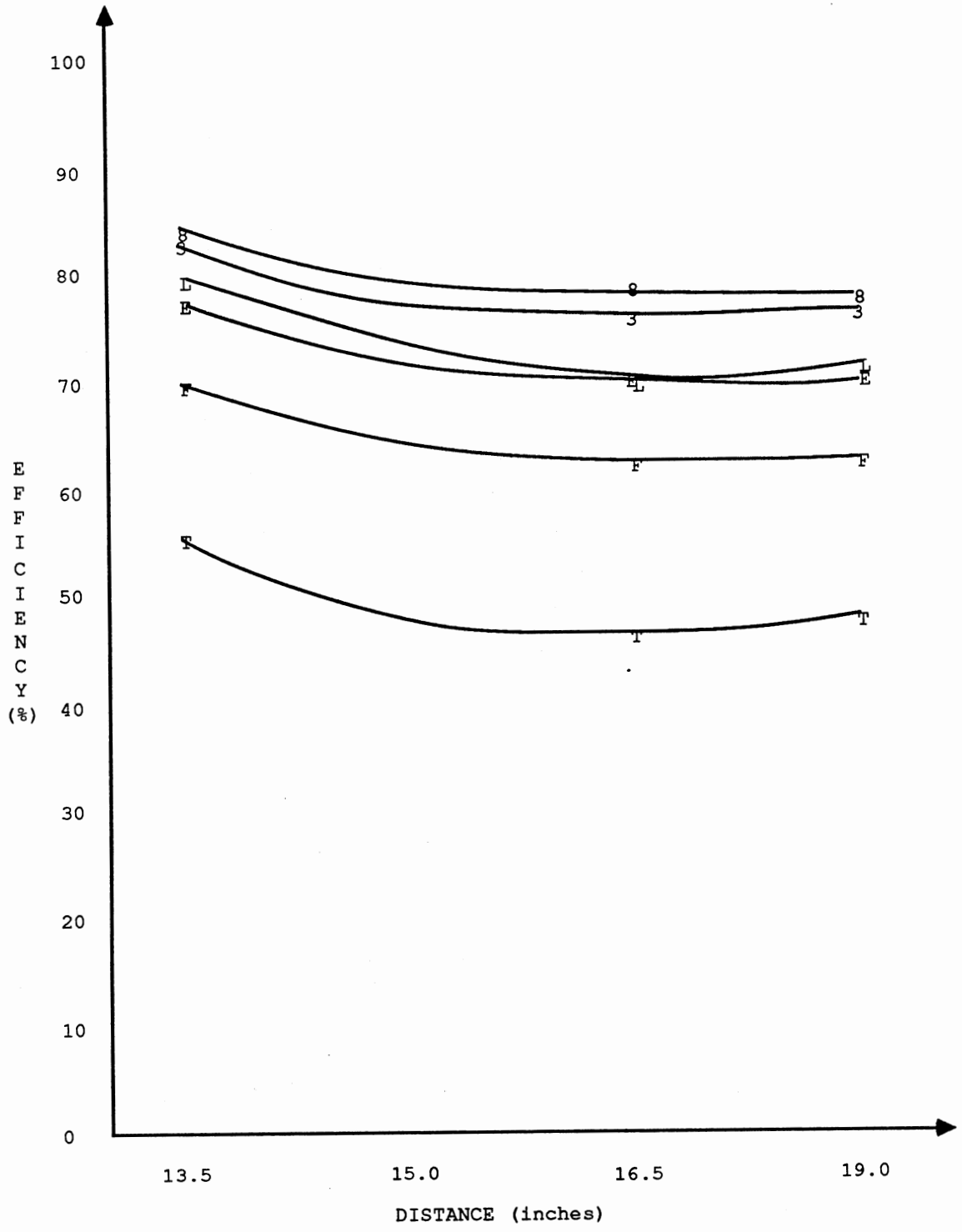


Figure 31. Focal Length Test Results (THR = 112)

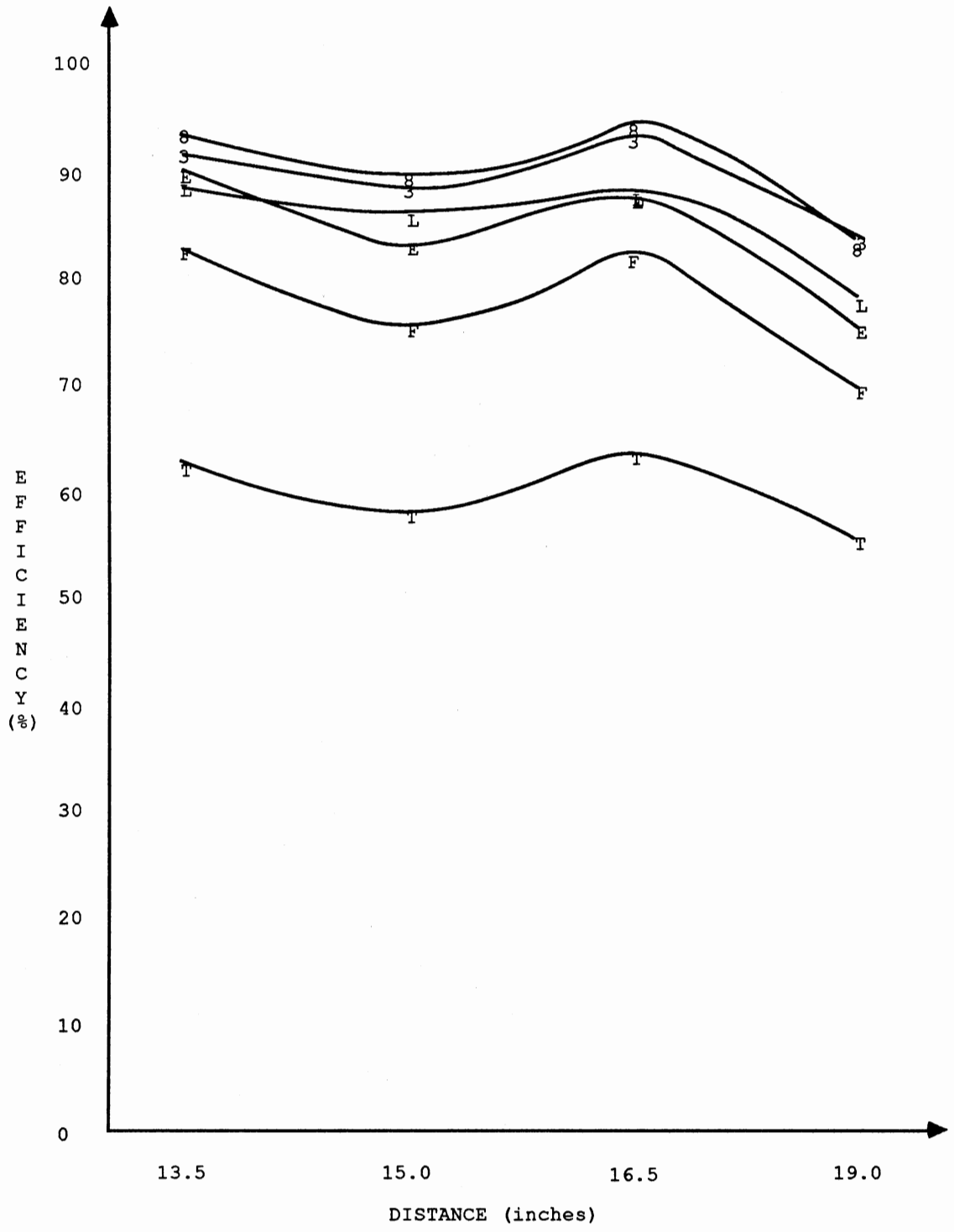


Figure 32. Focal Length Test Results (THR = 125)

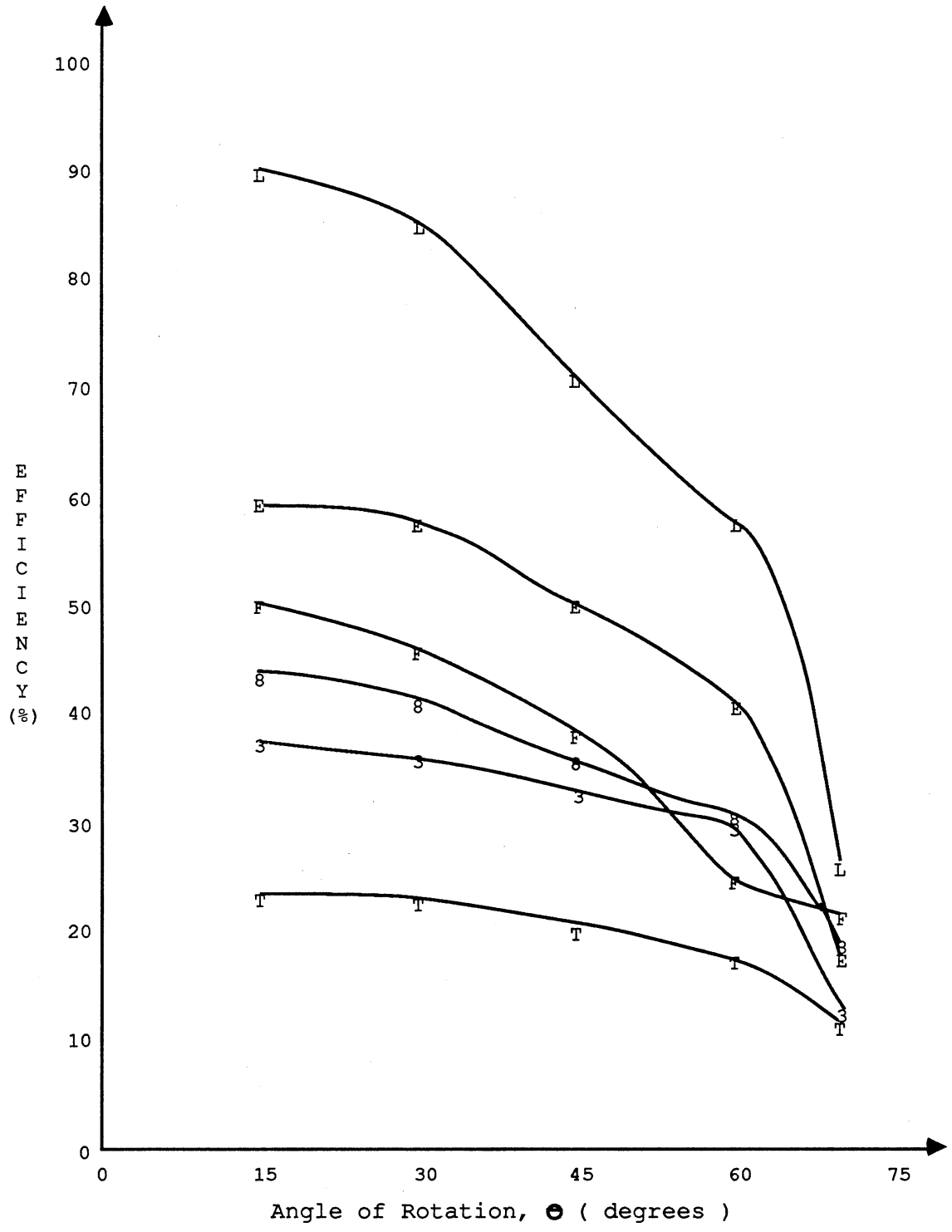


Figure 33. Positive Y-Axis Rotation Results

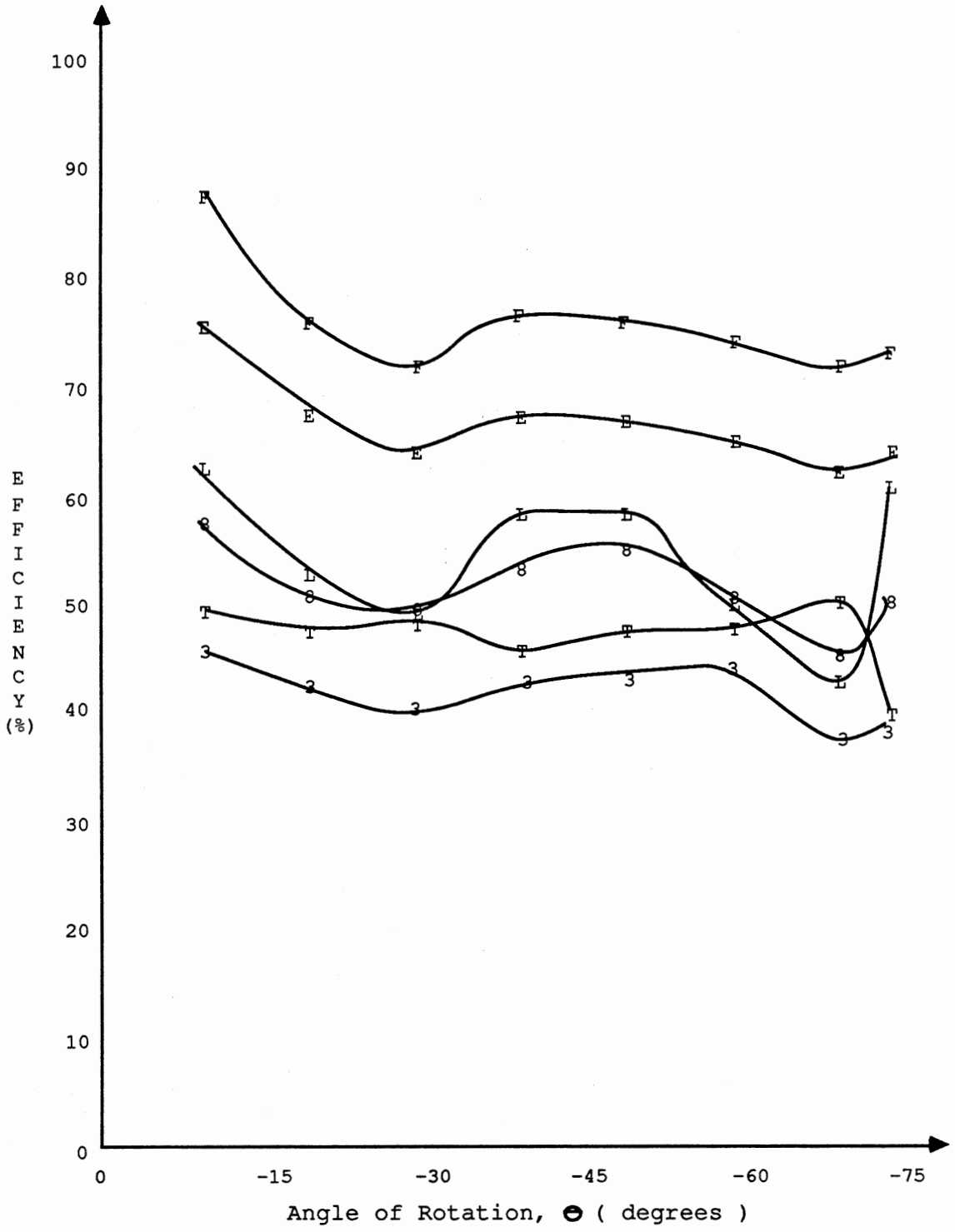


Figure 34. Negative Y-Axis Rotation Results

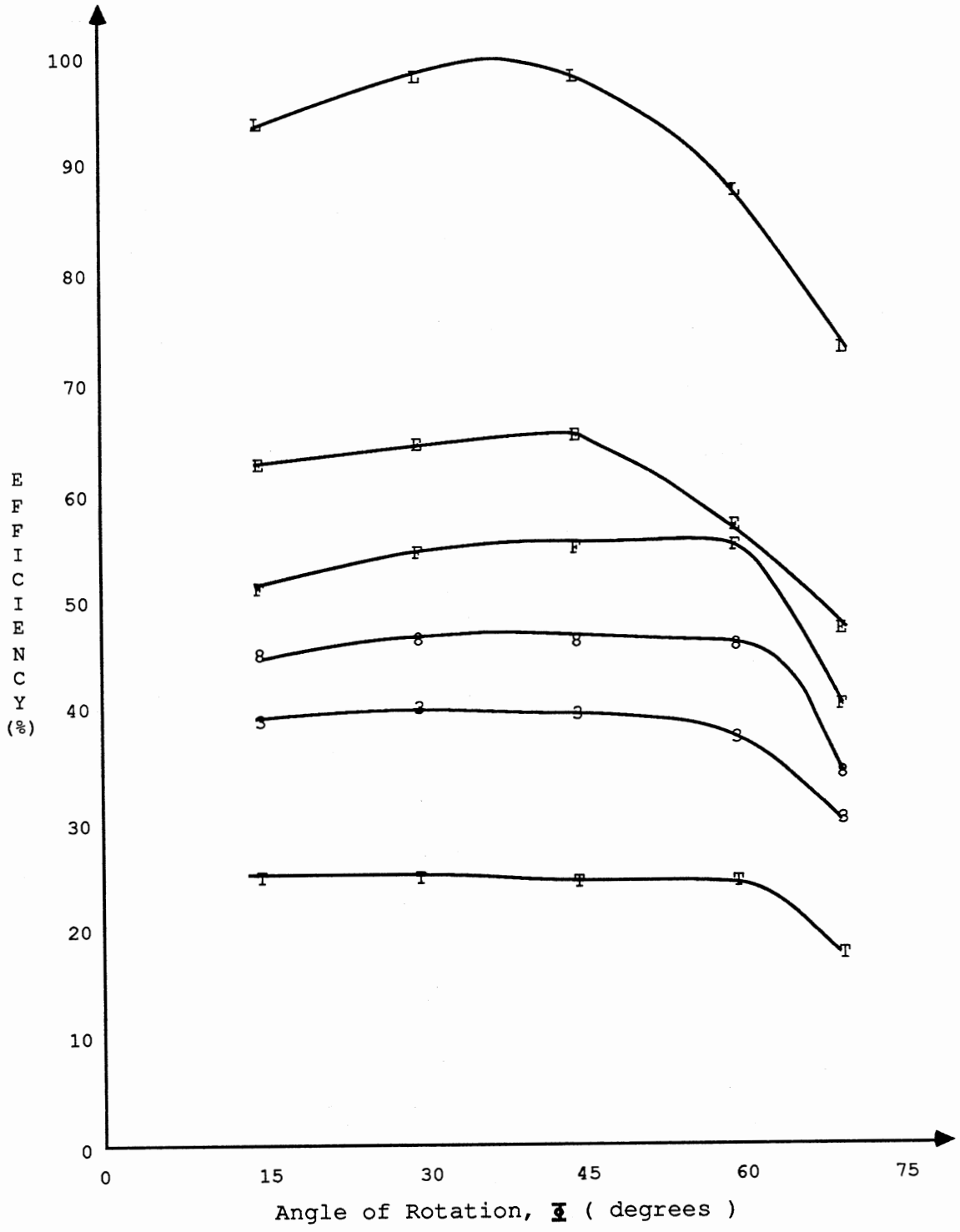


Figure 35. Positive X-Axis Rotation Results

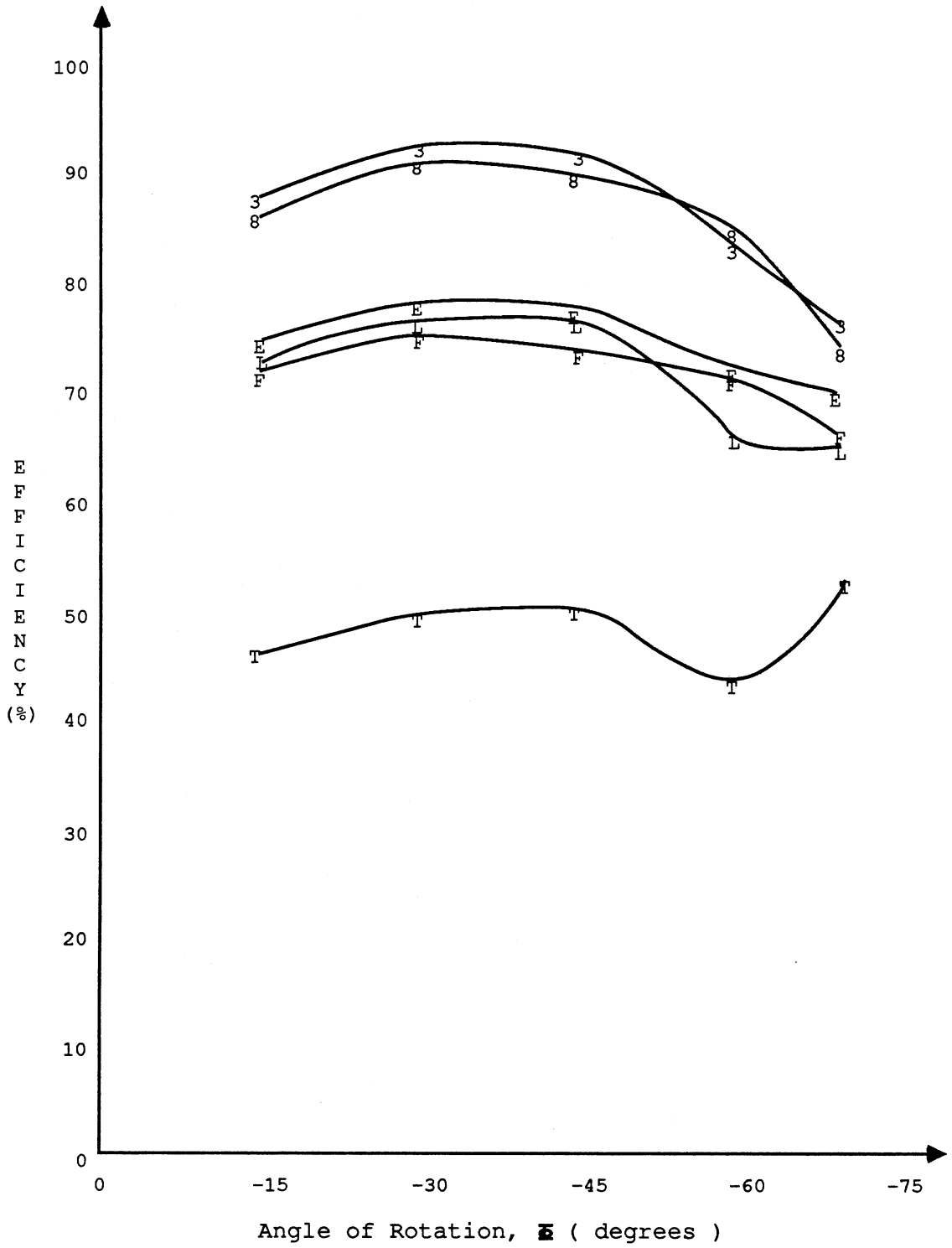


Figure 36. Negative X-Axis Rotation Results

efficiency for all templates versus the angle of rotation. The reconstructed character in Figures 33 and 35 is an L, the image in Figure 34 is an F, and in Figure 36, an 8. The actual data gathered during the computer runs is given in Table VI of Appendix B.

Examining the positive Θ rotation case in Figure 33, as Θ increases, the efficiency decreases. Also, at Θ=60 degrees the expansion algorithm had a fatal error which is not shown in the figure. The image was recaptured using different aperature and focus adjustments without moving the card, and the character was recognized correctly the second time. Thus, reproducability is a problem, apparently due to the physical sensitivities of the system. Figures 34, 35, and 36 show negative Θ, positive Φ, and negative Φ rotations respectively. As expected, all three figures show the efficiency decreases as the angle increases.

The four graphs also point out that the more dissimilar the captured character is to the stored templates, the more decisively the algorithm can recognize the character. For example, Figures 33 and 35 show that the L was correctly recognized by a significant efficiency margin over the next closest template (the E). This occurs because the L is unique among the six template characters. Likewise, the F was successfully recognized over the E in Figure 34, but by a smaller margin because the E and F are quite similar. This point is even more obvious for the negative Φ case in Figure 36, where the captured image is an 8, but was recognized at

most angles as a 3. Since the shape similarity between a 3 and an 8 is virtually the same as between an E and F, it was not clear at this point why the negative Φ case in particular was so unsuccessful. Additional data was taken at various Θ and Φ combinations to see if the poor recognition in the negative Φ cases would continue.

Random Rotation Results

The results of data collected at various Θ and Φ combinations up to $\pm$ 50 degrees are shown in Figure 37 and Table VII. The figure shows the results on an xy plane, as if looking down the z axis. The number in parentheses represents the efficiencies calculated for the correctly recognized characters shown next to them. The dark circles are cases in which the program could not complete the calculations. It stopped most frequently after the rotation algorithm. The efficiencies in the square are characters that were recognized incorrectly. For instance, at Θ = +35 and Φ = +20, the character E had an efficiency of 83, but the program recognized it as an F by calculating a higher efficiency of 86 for the F template.

As could be expected, the recognition algorithm is more frequently successful, with generally higher efficiencies, the closer the image is to the ideal, flat orientation (Θ = 0, Φ = 0). Also, the results for the negative Φ rotations were in general as good as the results for the positive Φ rotations. Hence, the negative Φ single axis rotation case

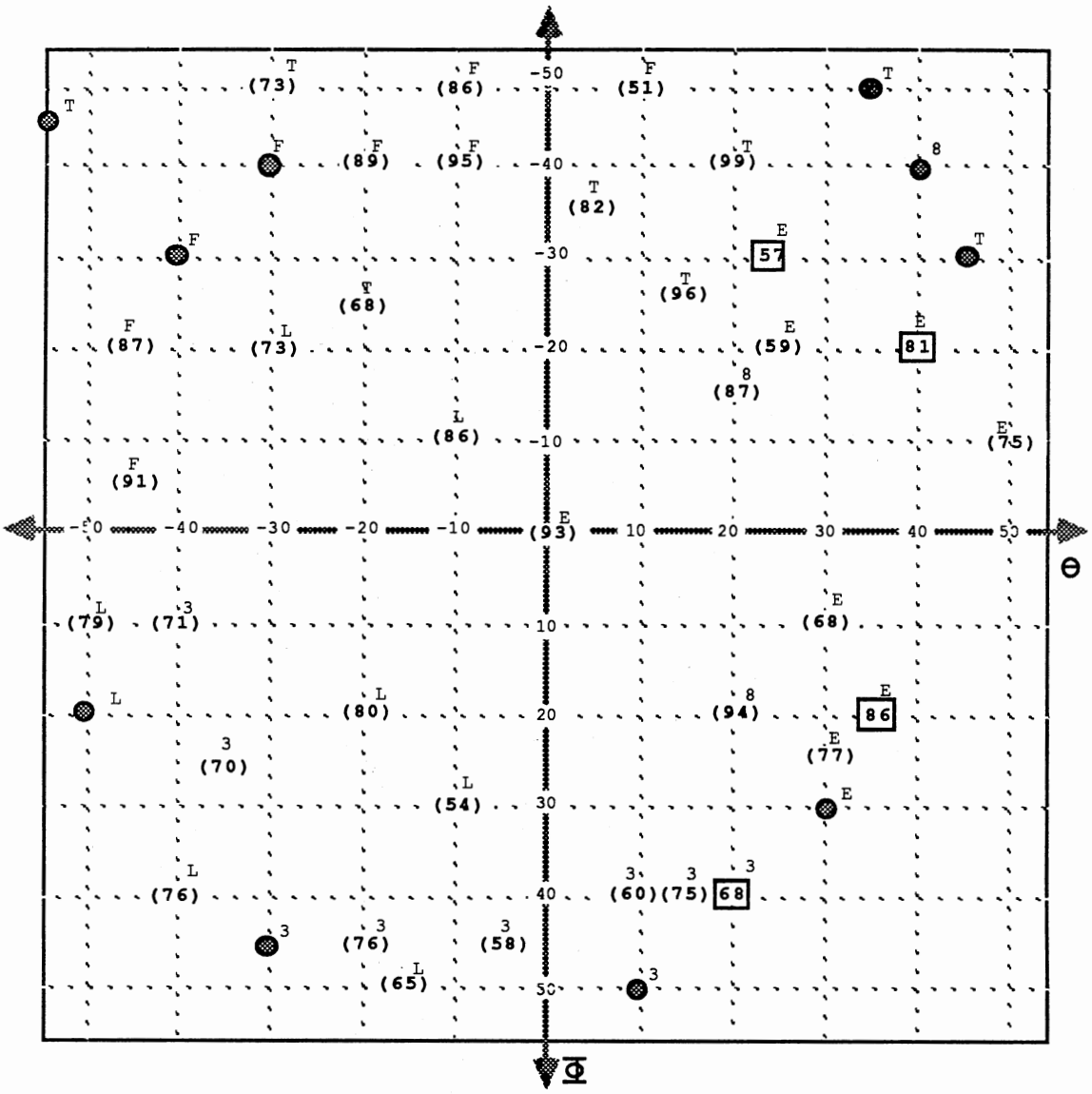


Figure 37. Random Angle Rotation Results

shown in Figure 36 was reexamined to find the cause of the recognition problem.

Examination of the efficiency calculation for the character 8 in Figure 36 pointed out a general inadequacy in the efficiency algorithm. The efficiencies are calculated by

counting the matching black pixels between the reconstructed image and each template, then dividing these counts by the total number of black pixels in each template. This introduces a problem since each template has a different total number of black pixels.

For instance, in the example cited earlier, the efficiency for the character E relative to the E template was only 83, and the character was incorrectly recognized as an F, with an efficiency of 86. The sum of the matching black pixels between the character E and the E template was 2,231, which is much higher than the 1,859 matching black pixels of the F template. However, the F template has only 2,159 total black pixels compared to the E's 2,676. These numbers illustrate that the matching black pixel count may be significantly higher for the correct template, but the calculated efficiency can still be higher for incorrect template. This problem only arises for similarly-shaped characters, such as E/F and 3/8 combinations, as supported by the data for other incorrectly recognized characters.

It is recommended that, as part of future research efforts, a more reliable recognition algorithm be developed. One method is to calculate the efficiency based on the total matching black and white pixels divided by the total number of pixels in the template, which is the same for all templates. Another approach is to use a different type of template matching or characteristic matching technique.

CHAPTER V

PARALLEL PROCESSING APPLICATIONS

The approach to vision system image recognition developed for this research was designed to be adaptable to a parallel processing system. Many of the algorithms have parallel structures which can be adapted to parallel processing. The obvious advantage to a parallel processing implementation is the significant increase in the vision system's time performance. The possibilities for parallel process applications range from a simple array processor to a uniquely designed vision system.

The first algorithms that would benefit from parallel processing are the scan techniques that define the card's edge parameters. The time efficiency of the search algorithm and the side angle and the corner calculations could be improved as well. Using parallel processing, a new scan technique could be implemented to use a series of scan lines to find multiple edge intersect points, thus producing more accurate angle and corner results.

The threshold and nearest-neighbor filters used in this research could also be enhanced by parallel processing. Since both of these filters are performed on a pixel by pixel basis, the processing time could be reduced by a factor

approximately equal to the number of processors used in the system. However, a problem would arise for the neighborhood averaging filter, which uses information from surrounding pixels as a basis for its decision-making process. For that case, the image would be broken into regions, as opposed to rows or columns, and each processor could be assigned a given region for its processing.

The expansion algorithms are adaptable to parallel processing because they are independent row and column algorithms. The x-expansion algorithms calculate the expansion ratios and the bit-fill patterns in a parallel structure, and perform the actual expansion in a parallel row format. The y-expansion algorithms use a similar column format. Thus, by assigning each processor first the rows and then the columns, the expansion algorithm performance could again be reduced approximately by the number of processors.

The final set of algorithms that could benefit from parallel processing are the recognition algorithms. Once the image has been reconstructed, the number of processors would directly affect the comparison time required in the template matching scheme. By assigning each processor to a particular template and comparing its template to a copy of the reconstructed image, the comparison time would be significantly reduced.

CHAPTER VI

SUMMARY AND CONCLUSIONS

The primary goal of this research is to develop a vision system approach to recognize severely skewed alphanumeric characters. To implement this concept on the available equipment, a VAX 11/750 and a COMTAL Image Processing System, a FORTRAN program was developed to recognize a single black character on a square white card.

In general, the vision system algorithms were successful in reconstructing and recognizing the skewed characters. Specifically, the algorithms were able to recognize a character tilted 75 degrees in a single-axis rotation case with a 73% efficiency. They were also able to recognize a character having angles of rotation Θ=-30 and Φ=-50 with a 73% efficiency. However, it was shown that dissimilar characters were recognizable with greater efficiency and reliability compared to characters with similar features or shapes, such as E's and F's. Although the efficiency of the reconstruction algorithms decreased as the card's degree of tilt increased, the programs were still able to recognize the character correctly, with few exceptions.

Poor captured image quality can cause problems in producing consistent results for similar images. It can also

cause the reconstructed images to have significant digitizing errors. One cause of the poor captured image quality is a problem inherent in the system hardware. The quality of the captured image is greatly dependent upon the lighting, focus, and aperture adjustments of the camera system. The captured image can be too bright or too dark depending on the positioning of the lighting and the camera settings. The sensitivity of the focus adjustment coupled with the poor lighting causes difficulty in capturing sharp images.

The main conclusion to be drawn from this research is that since the vision system algorithms can successfully recognize skewed characters, this approach can be applied to industrial applications. However, in its current form, the recognition algorithm is slightly unreliable and the programs are too inefficient for most real-time applications. The recognition problem can be alleviated by modifying the template-matching process to calculate an efficiency based on the white and black pixels common to the image and each template, instead of based on the common black pixels only. By using state-of-the-art equipment and a multi-processor system, the time efficiency of the algorithms could be significantly improved. It is recommended that further investigation into the use of these algorithms in a parallel processing system be pursued.

APPENDIXES

APPENDIX A - FLOW CHARTS

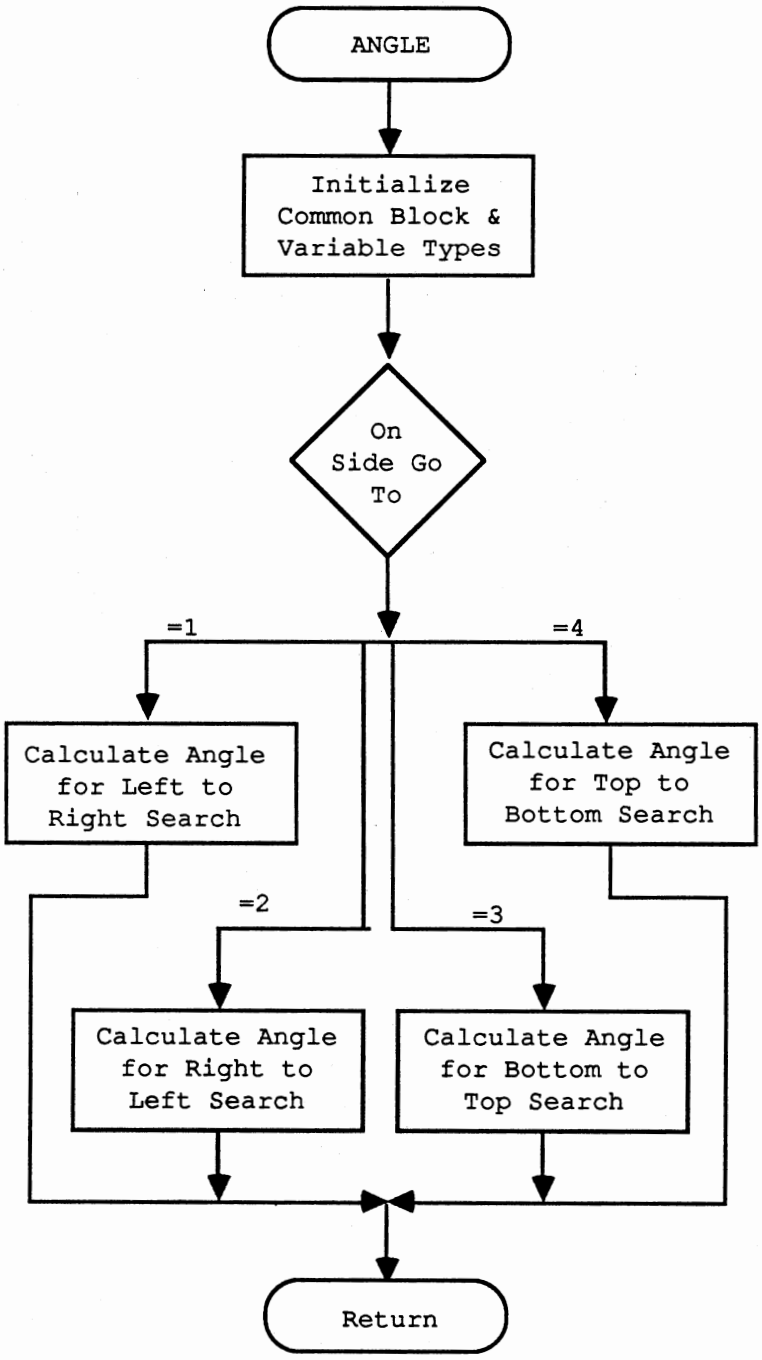


Figure 38. Calculate Side Angles Flow Chart

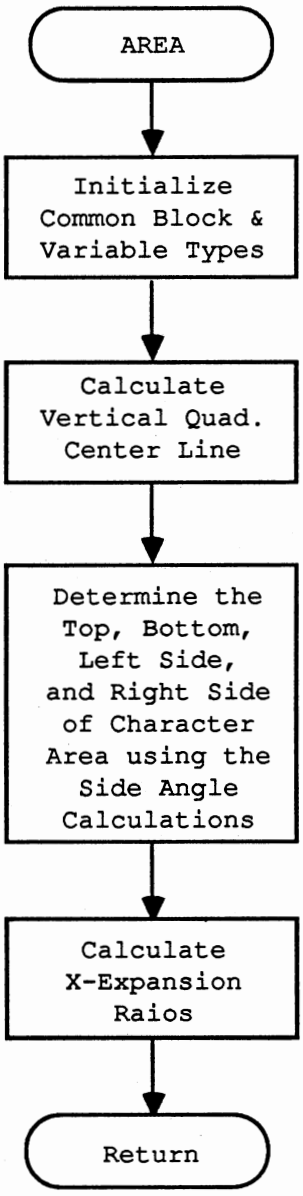


Figure 39. Calculate Area Shape Flow Chart

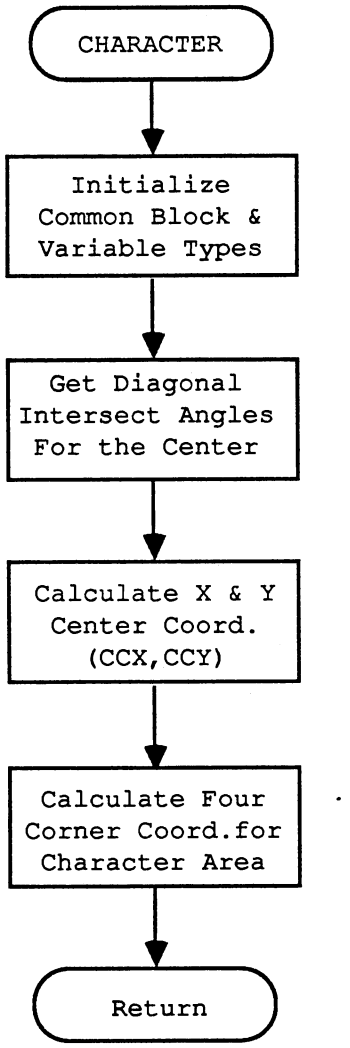


Figure 40. Calcuate Character Area Flow Chart

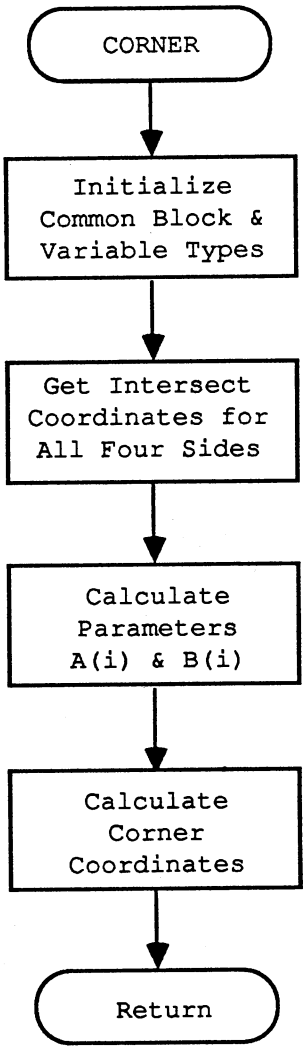


Figure 41. Calculate Corner Coordinates Flow Chart

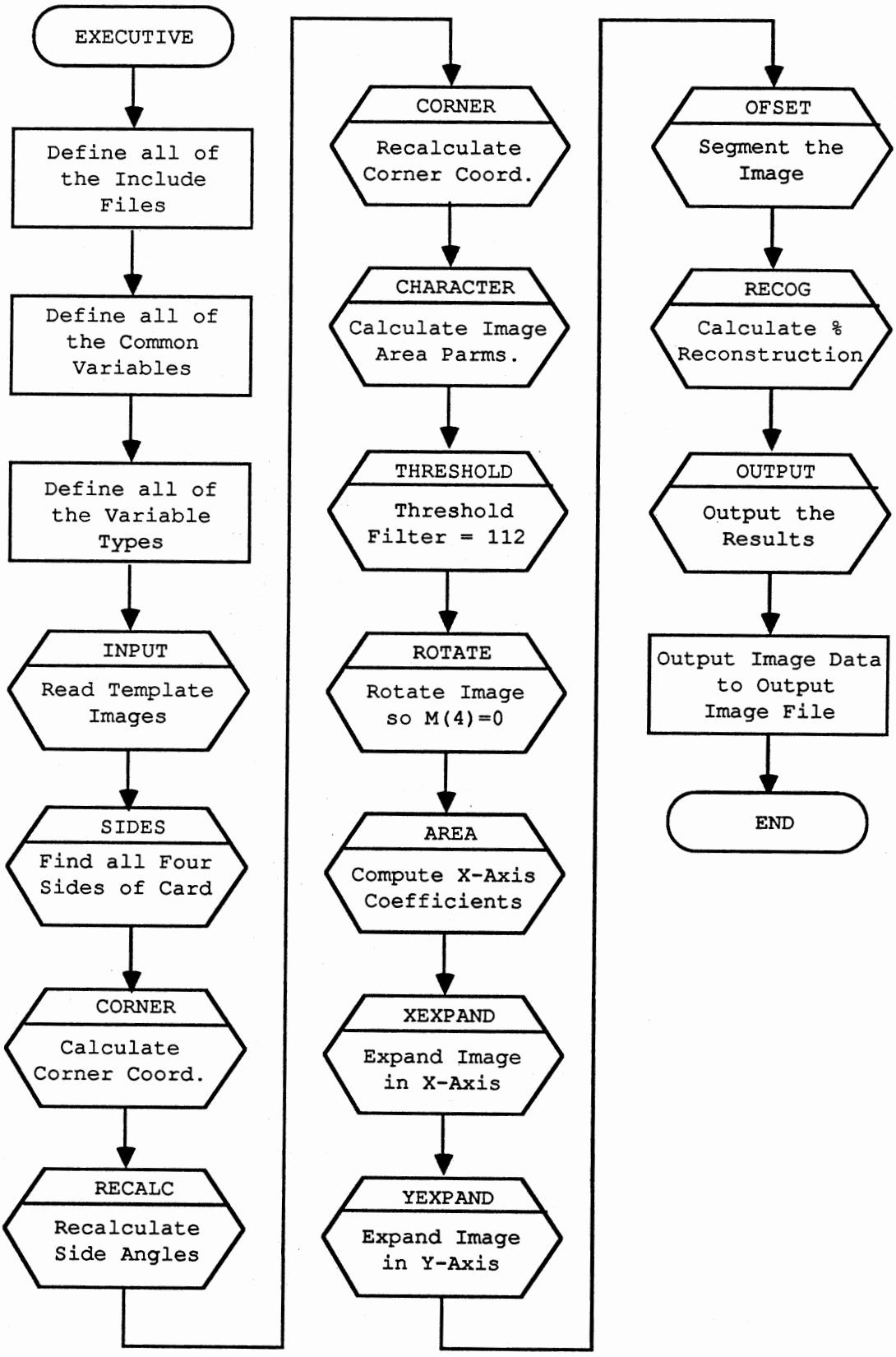


Figure 42. Executive Flow Chart

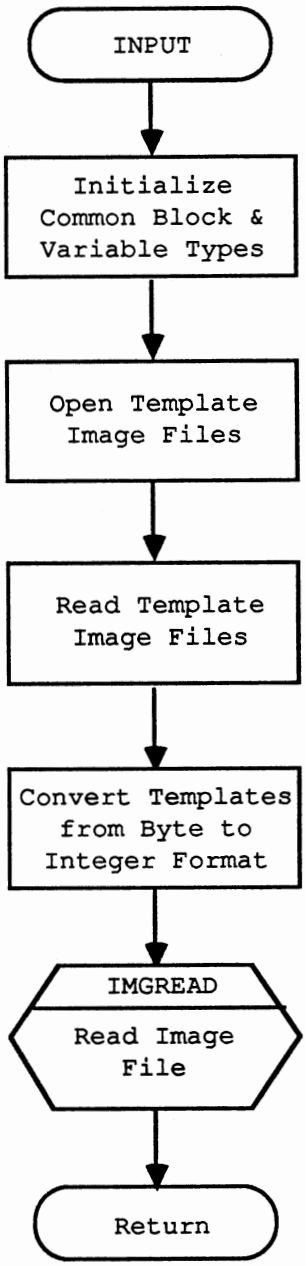


Figure 43. Input Templates Flow Chart

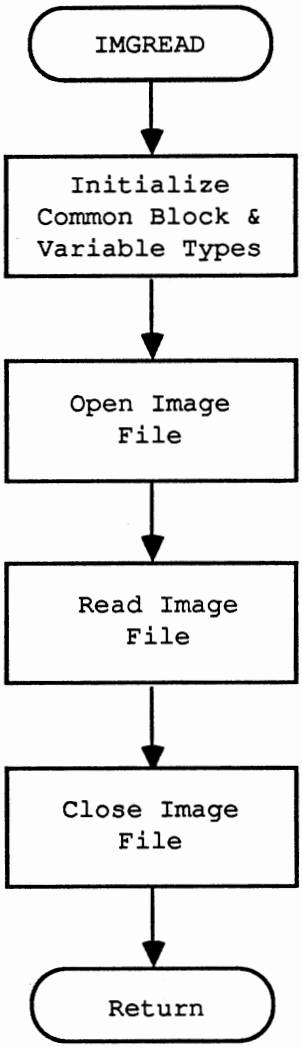


Figure 44. Read Image Flow Chart

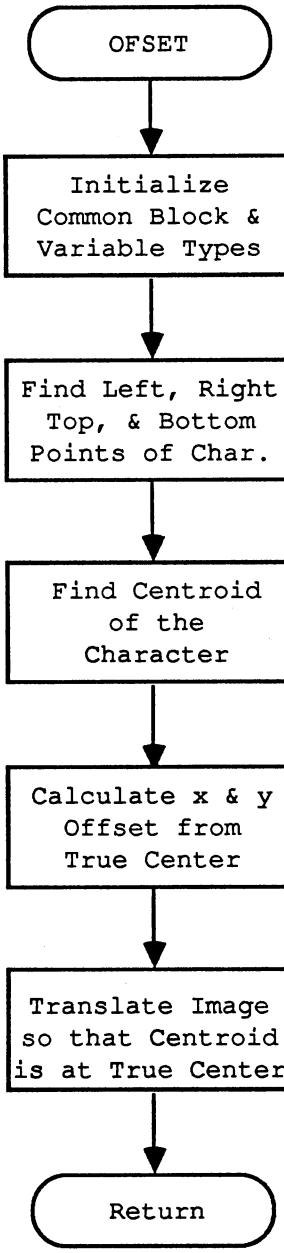


Figure 45. Offset Calculation Flow Chart

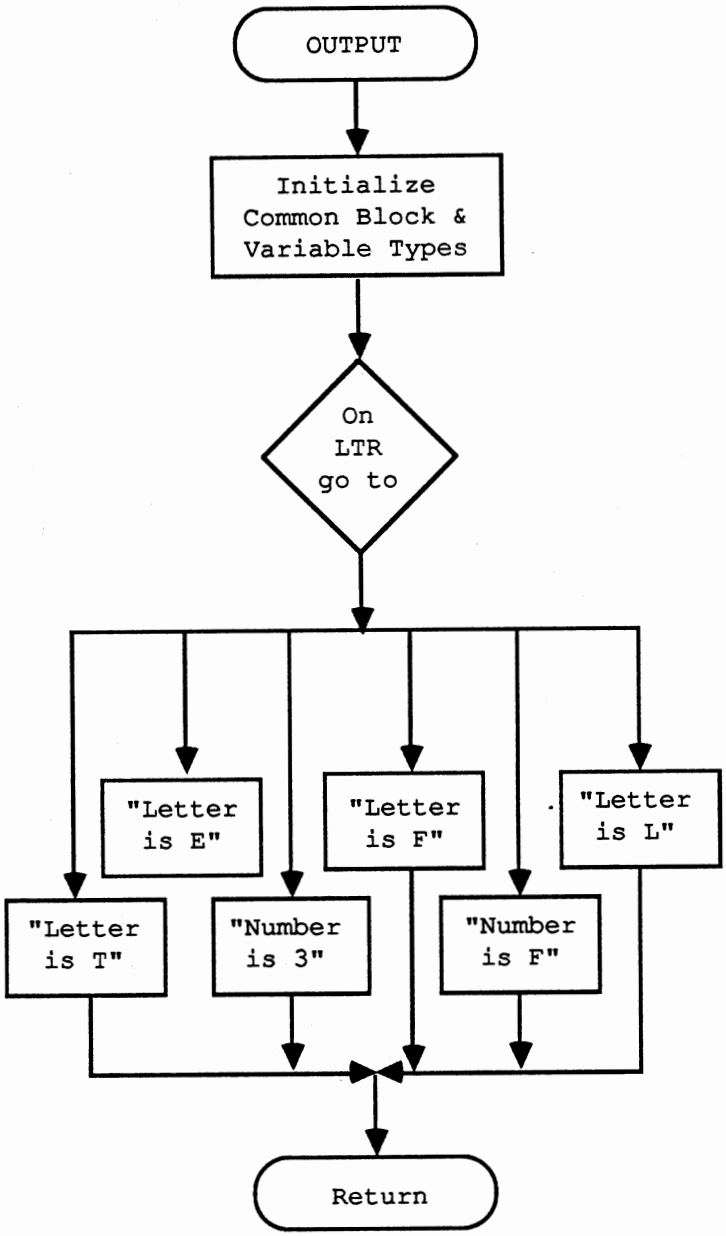


Figure 46. Output Results Flow Chart

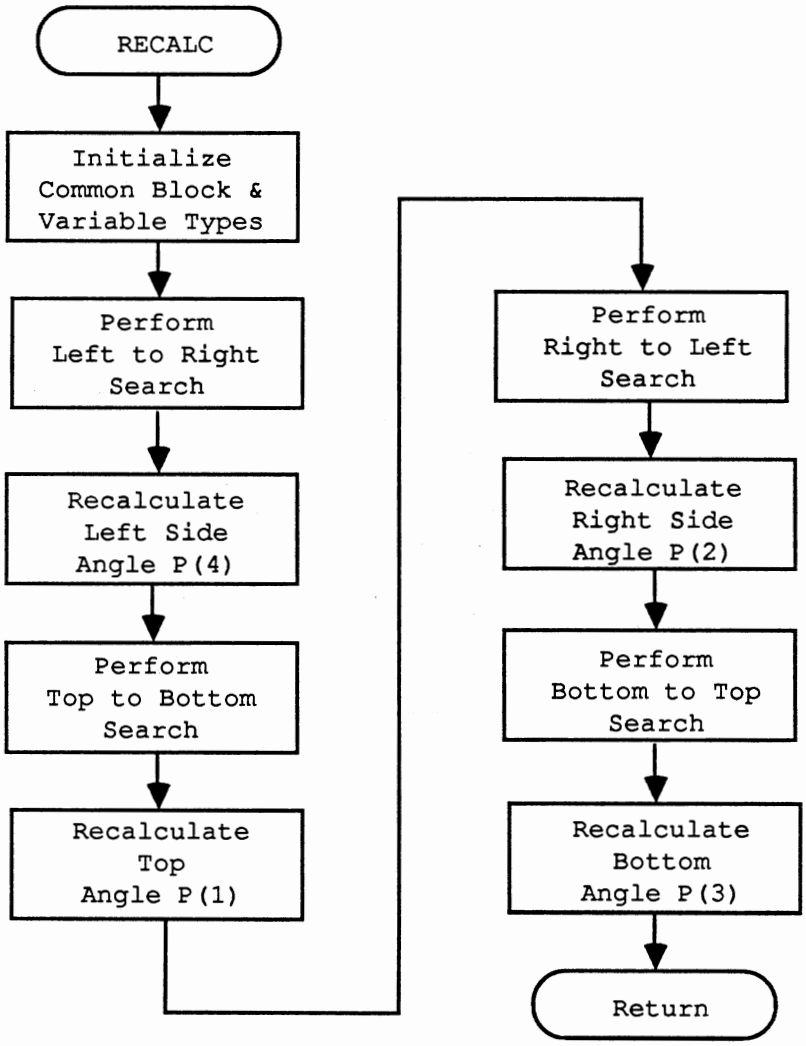


Figure 47. Recalculate Corner Coordinates Flow Chart

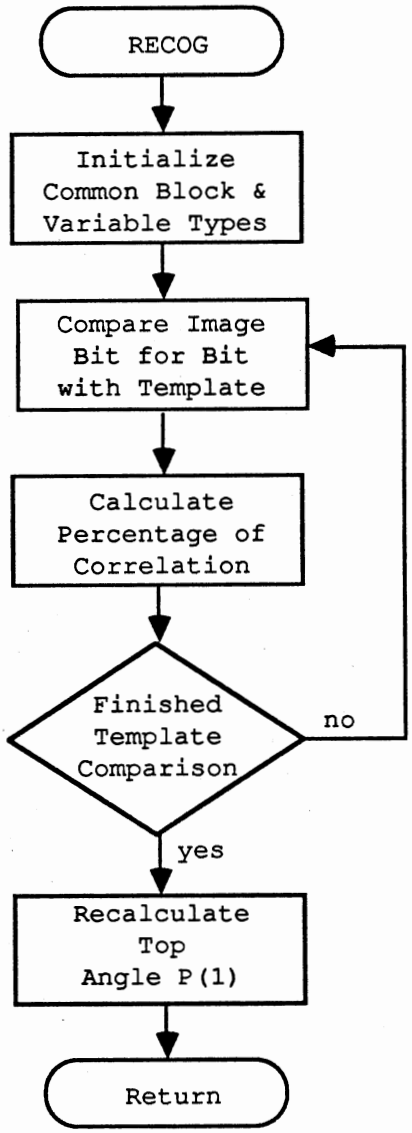


Figure 48. Recognize Character Flow Chart

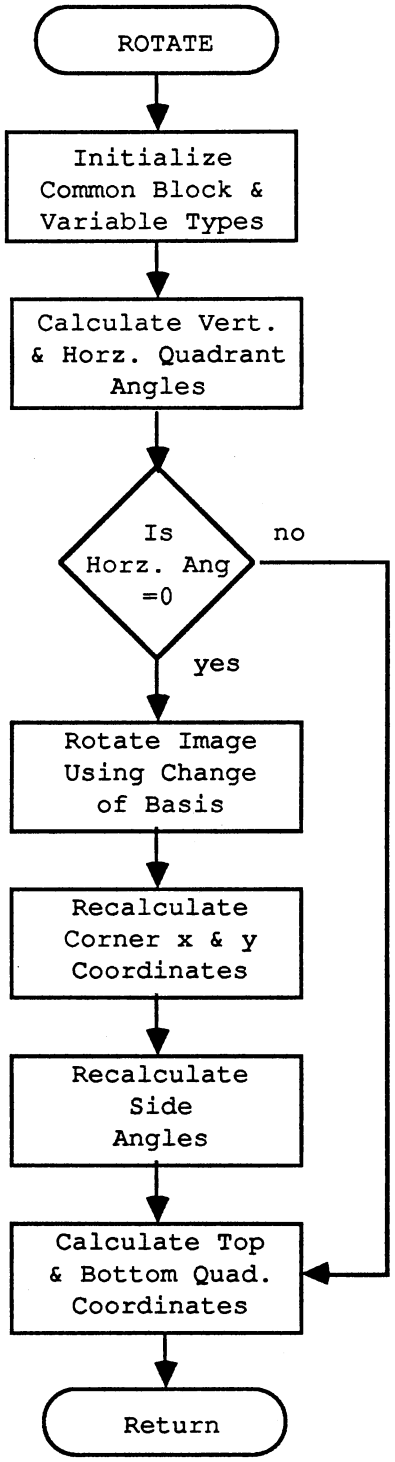


Figure 49. Rotate Image Flow Chart

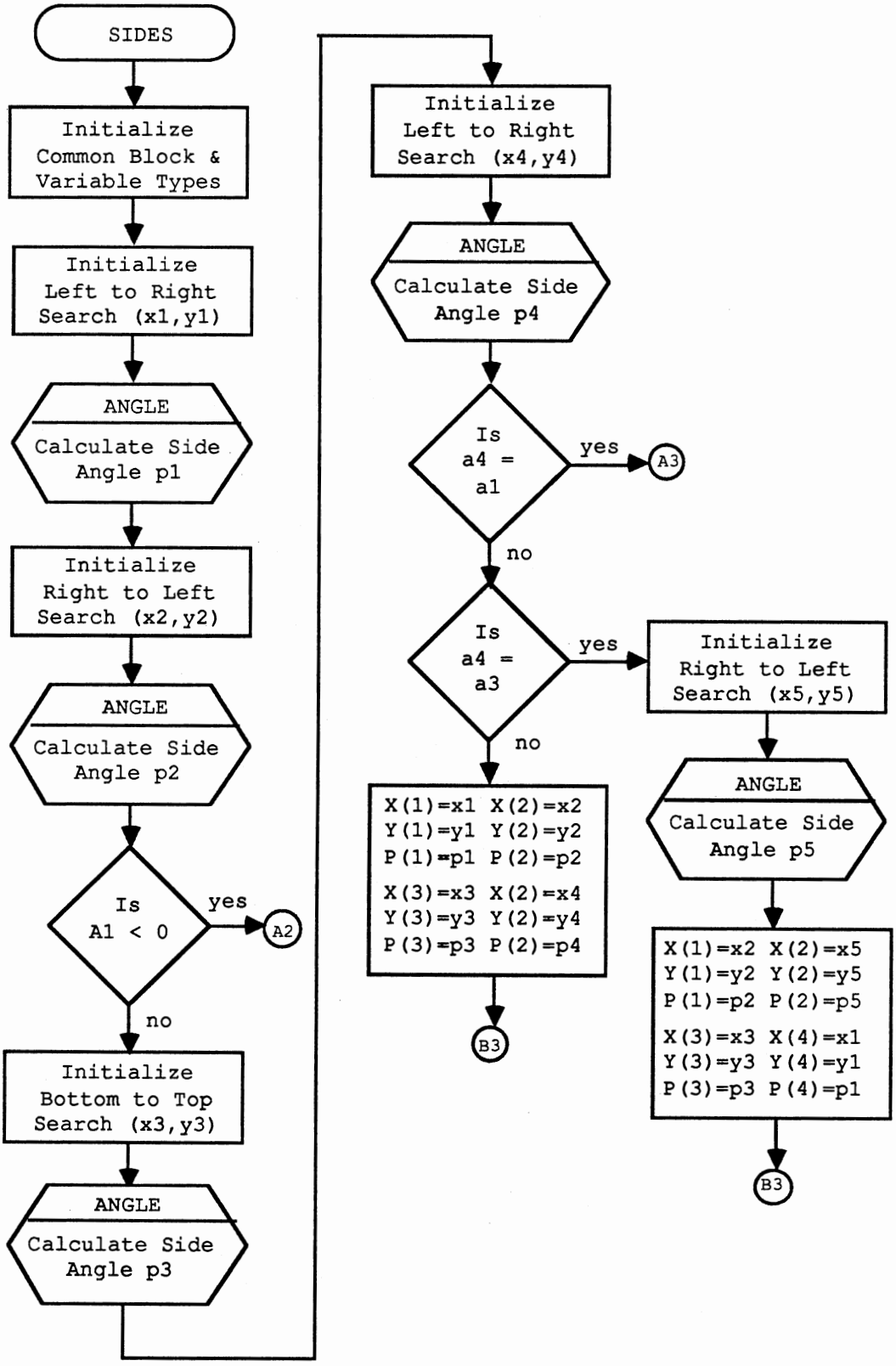


Figure 50. Find Sides Flow Chart (1 of 3)

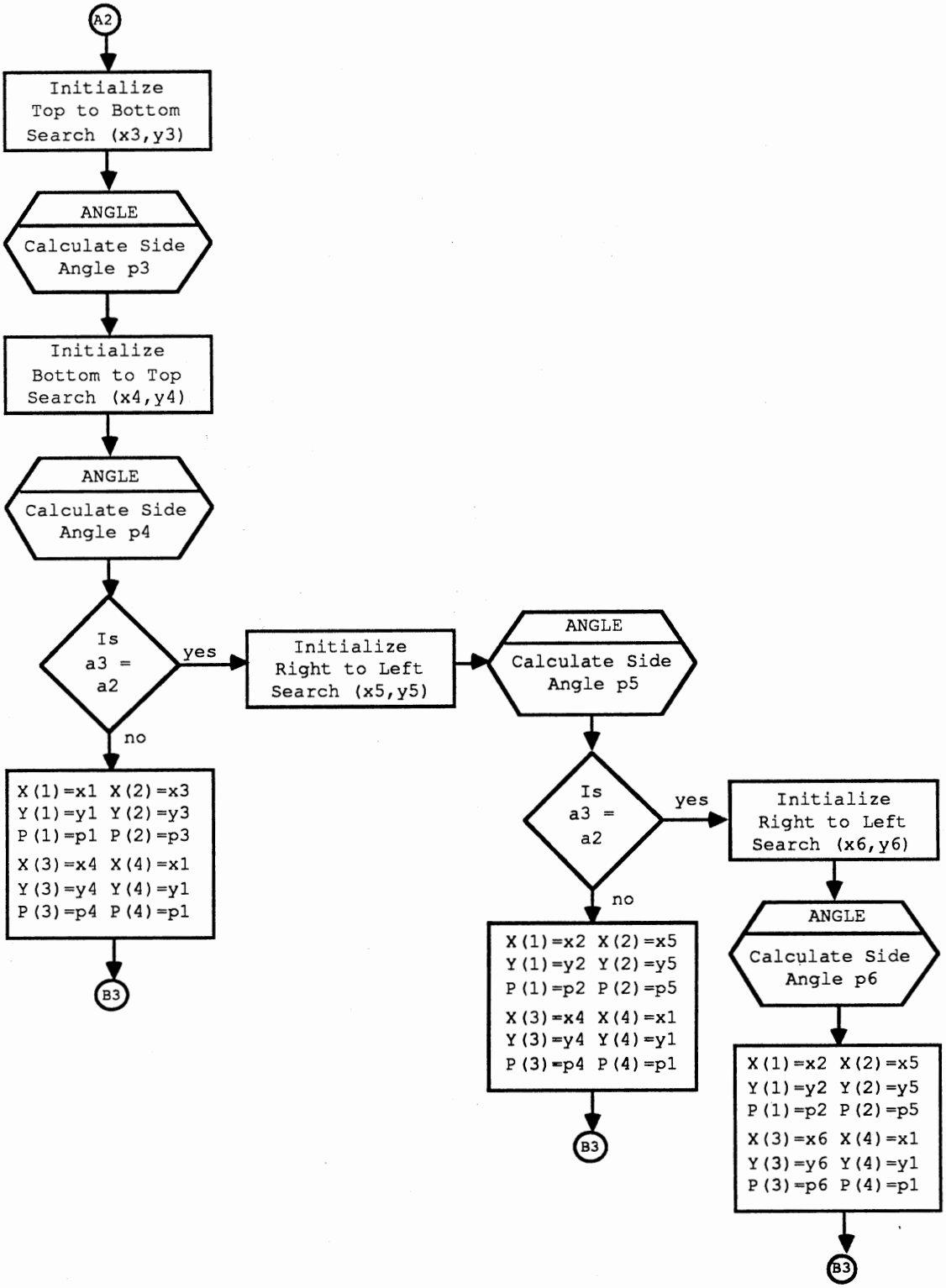


Figure 51. Find Sides Flow Chart (2 of 3)

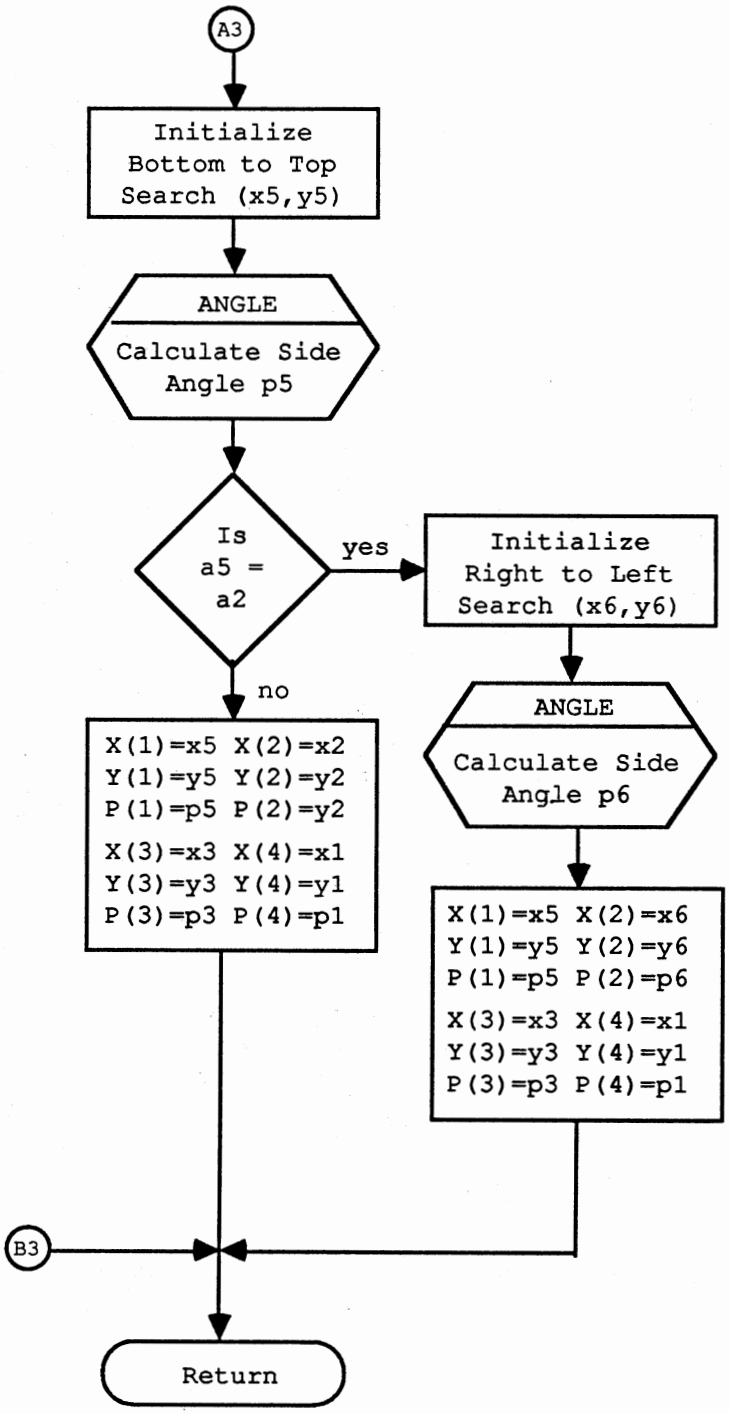


Figure 52. Find Sides Flow Chart (3 of 3)

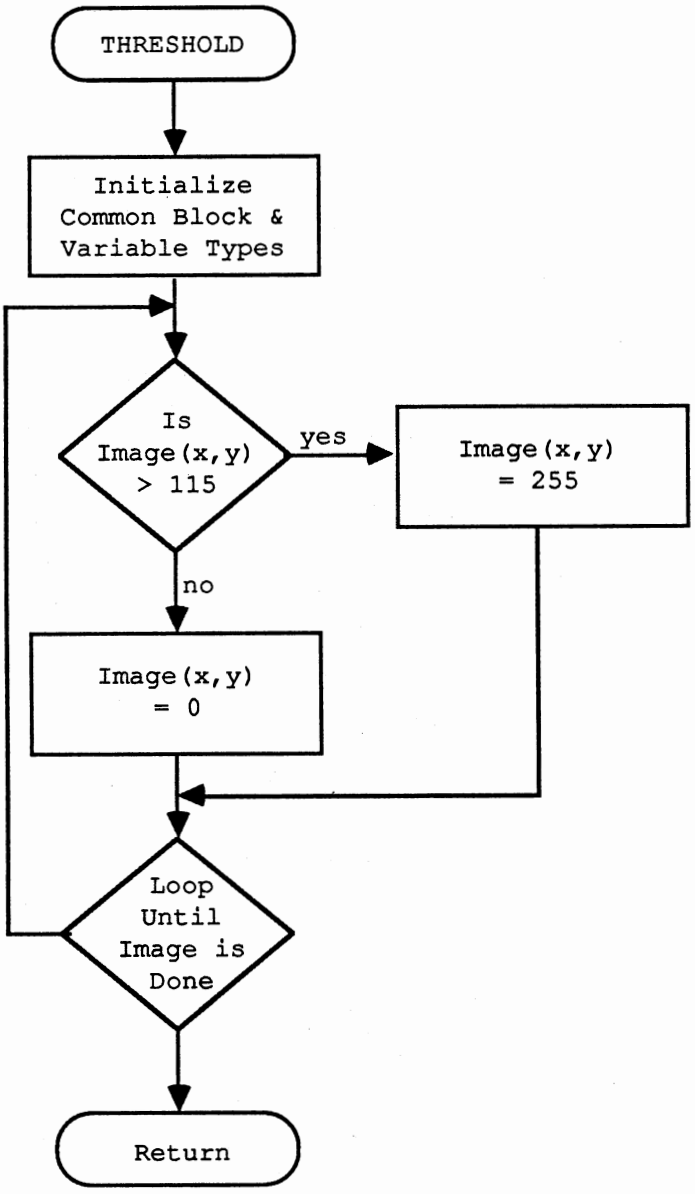


Figure 53. Threshold Flow Chart

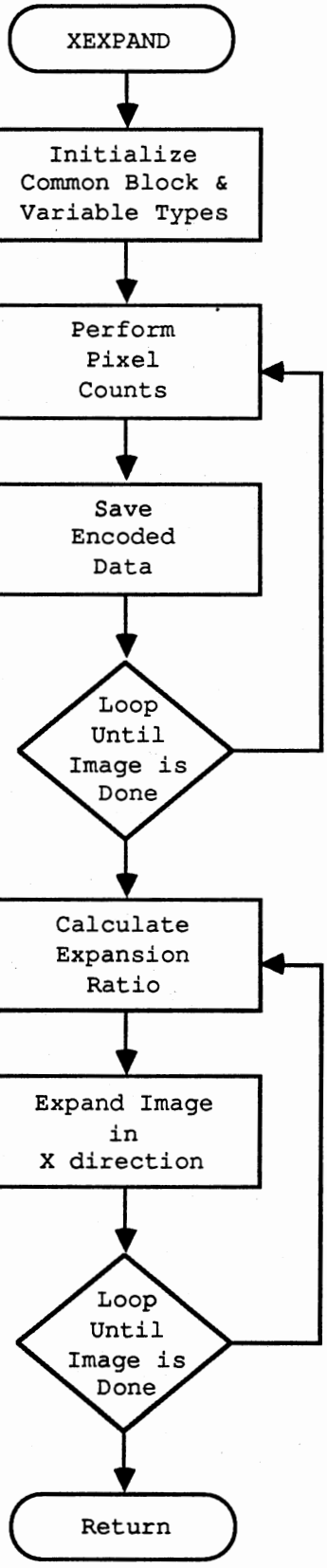


Figure 54. X-Expansion Flow Chart

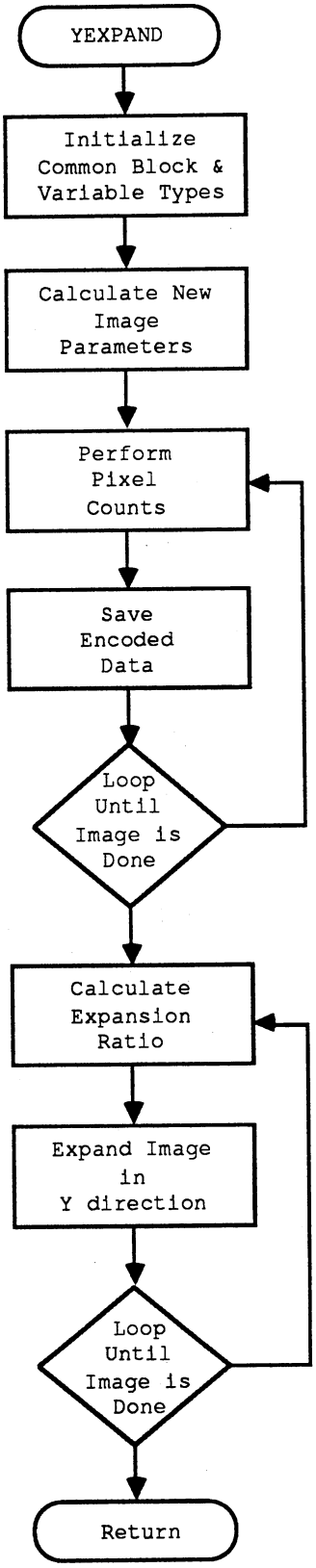


Figure 55. Y-Expansion Flow Chart

APPENDIX B - TEST DATA

TABLE IV

THRESHOLD TEST DATA

LTR	THR	Efficiency (%) E	F	L	T	3	8
3	75	**	**	**	**	**	**
	100	35.7	26.8	33.7	30.1	49.3	45.6
	112	41.8	32.0	38.7	36.3	59.8	54.1
	125	47.5	36.6	45.4	40.6	64.5	58.5
	138	52.8	41.7	49.3	46.0	71.8	64.3
	150	57.4	46.0	52.0	51.2	77.5	69.6
	165	58.3	51.6	71.9	57.7	72.8	67.7
T	75	29.7	32.3	14.5	77.5	23.0	20.6
	100	36.2	37.0	20.3	88.9	29.3	26.9
	112	39.2	39.3	22.9	93.2	32.2	29.9
	125	44.1	44.1	26.7	99.1	38.1	35.3
	138	47.2	50.5	31.7	99.9	41.9	39.8
	150	50.0	49.3	36.0	71.9	74.6	59.5
L	75	26.0	11.9	33.5	0.1	16.6	16.2
	100	40.3	25.2	56.4	16.3	27.4	27.6
	112	48.0	29.9	67.6	19.6	32.5	32.6
	125	47.9	29.4	68.2	21.1	35.0	34.3
	138	55.6	38.6	80.4	25.4	40.1	40.7
	150	16.5	12.0	12.6	12.1	37.2	31.6

TABLE V

FOCAL LENGTH TEST DATA

LTR	THR	DIS	Efficiency (%)					
			E	F	L	T	3	8
8	100	13.5	71.6	62.4	73.1	47.9	74.4	76.7
		15.0	77.2	68.4	79.3	51.4	81.9	84.0
		16.5	80.9	73.0	81.0	53.7	84.3	87.1
		19.0	62.3	56.0	64.3	41.1	69.3	71.3
	112	13.5	78.4	69.6	79.9	54.9	82.4	83.9
		15.0	71.6	62.4	73.1	47.9	74.4	76.7
		16.5	70.0	62.2	69.0	46.5	76.0	78.2
		19.0	70.3	62.8	71.7	48.5	77.8	78.5
	125	13.5	89.6	82.3	89.8	62.5	91.2	93.2
		15.0	83.2	75.3	85.7	58.2	87.9	89.5
		16.5	88.5	81.9	88.7	62.9	92.7	94.2
		19.0	75.2	69.1	78.3	56.0	83.3	83.1

TABLE VI

SINGLE AXIS ROTATION DATA

LTR	Θ	Φ	Efficiency (%)					
			E	F	L	T	3	8
F	0	-10	75.5	87.2	62.5	49.3	47.1	57.4
	0	-20	67.5	76.1	52.3	47.8	42.4	50.7
	0	-30	64.5	71.8	48.9	48.0	40.0	49.2
	0	-40	68.4	77.1	57.9	45.9	42.7	53.8
	0	-50	67.1	75.7	58.0	47.6	43.0	55.1
	0	-60	65.4	73.9	49.9	48.3	44.1	50.2
	0	-70	21.7	25.8	32.4	18.3	19.2	26.7
	0	-70	19.0	21.8	27.5	16.3	18.7	24.3
	0	-70	62.4	72.2	42.6	50.4	37.8	45.8
	0	-75	64.4	72.9	61.5	39.8	38.8	50.3
L	15	0	62.9	51.3	94.3	24.5	39.3	45.4
	30	0	64.4	54.7	98.0	24.6	40.1	47.2
	45	0	65.5	55.2	98.1	24.2	39.7	46.2
	60	0	58.3	55.9	87.3	23.9	38.4	46.4
	70	0	48.5	40.6	72.9	18.7	29.8	33.9
	75	0	**	**	**	**	**	**
L	0	15	59.0	50.0	89.6	22.7	38.1	43.4
	0	30	57.5	45.5	84.5	22.4	36.1	41.3
	0	45	49.7	37.8	70.5	20.0	32.4	35.8
	0	60	38.6	22.3	60.5	18.5	29.6	31.2
	0	60	**	**	**	**	**	**
	0	70	18.2	21.2	27.1	11.6	12.2	18.4
8	-15	0	74.4	71.0	72.8	46.4	86.9	85.9
	-30	0	78.0	74.7	75.9	49.2	92.2	90.2
	-45	0	77.5	73.1	76.2	50.0	91.3	89.7
	-60	0	71.7	70.8	65.6	43.0	82.9	84.2
	-70	0	**	**	**	**	**	**
	-70	0	**	**	**	**	**	**
	-70	0	69.8	66.1	64.4	52.6	76.4	73.1

TABLE VII.

RANDOM ANGLE ROTATION DATA

LTR	Θ	Φ	Efficiency (%)					
			E	F	L	T	3	8
L	-50	10	49.9	51.4	79.3	22.3	34.3	41.4
F	-45	-20	69.6	87.4	65.5	42.4	49.3	58.9
F	-45	-20	77.3	91.2	70.0	48.8	49.6	60.9
3	-40	10	56.3	46.3	48.1	54.3	71.4	62.4
L	-40	40	49.9	46.7	75.5	24.4	31.5	38.7
3	-35	25	49.8	44.7	50.1	42.8	69.5	53.8
T	-30	-50	39.7	39.6	25.5	72.9	47.7	36.4
	-30	-20	41.8	42.1	66.1	20.6	28.9	35.3
F	-20	-40	75.5	89.3	71.6	47.3	49.1	60.3
T	-20	-25	31.2	33.6	18.1	68.2	25.9	24.3
L	-20	20	55.3	42.7	79.5	22.1	35.3	39.6
3	-20	45	48.3	45.4	42.1	38.1	76.0	60.1
L	-15	50	43.7	44.4	65.3	15.5	24.4	33.4
F	-10	-50	74.5	86.4	72.9	59.8	49.2	58.9
F	-10	-40	78.4	95.4	70.6	48.9	49.3	59.5
L	-10	-10	57.8	48.4	86.3	22.0	35.1	41.3
L	-10	30	33.5	32.3	54.4	16.2	25.4	29.8
3	-5	45	43.9	42.3	39.5	46.1	58.2	44.0
E	0	0	92.6	85.4	89.3	57.8	65.5	70.5
T	5	-35	59.7	72.6	57.6	81.8	41.0	47.8
F	10	-50	47.0	50.9	25.9	46.1	29.7	34.6
3	10	40	40.8	42.7	34.6	41.3	59.6	44.3
T	15	-35	51.2	52.8	30.5	95.6	67.6	57.2
3	15	40	56.4	47.3	44.6	56.0	74.9	62.1
T	20	-40	70.0	77.8	65.3	99.7	75.4	72.9
8	20	-15	76.5	72.3	73.8	50.5	87.4	87.2
8	20	20	81.4	78.3	81.5	49.5	93.6	94.3
8	20	40	59.1	52.1	43.4	51.5	75.4	67.6
E	25	-30	57.2	56.5	60.9	49.1	44.3	49.0
E	25	-20	58.7	49.1	52.3	52.7	46.1	45.0
E	30	10	68.2	60.3	58.3	49.0	50.8	52.8
E	30	25	76.7	69.5	63.5	60.3	56.3	57.2
E	35	20	83.4	86.1	83.5	54.1	64.9	68.7
E	40	-20	77.0	70.8	80.7	50.6	55.6	60.8
E	50	-10	74.8	68.7	74.8	52.2	54.6	58.6

SELECTED BIBLIOGRAPHY

(1) Bower, Frank, "CCD Fundamentals." Fairchild Charged-Coupled Device Catalog (1984), pp. 121-136.

(2) Amelio, Gilbert F., "Charge-Coupled Devices." Fairchild Charged-Coupled Device Catalog (1984), pp. 126-137.

(3) Gonzalez, Rafael C. and Paul Wintz. Digital Image Processing Massachusetts: Addison-Wesley Publishing Company, Inc., 1983.

(4) Oppenheim, Alan V. and Ronald W. Schafer. Digital Signal Processing New Jersey: Prentice-Hall, Inc., 1975.

(5) Schowengerdt, Robert A., Techniques for Image Processing and Classification in Remote Sensing. New York: Academic Press, Inc., 1983.

(6) Papoulis, Athanasios. Probability, Random Variables, and Stochastic Processes. 2nd edition New York: McGraw-Hill Book Company, 1984.

(7) Scher, Ann, Velasco, Flavio R. Dias, Rosenfeld, Azriel, "Some New Image Smoothing Techniques." IEEE Transactions on System, Man and Cybernetics, (March, 1980), pp. 153-158.

(8) Hoyer, A., Schlindwein, M. "Digital Image Enhancement." Philips tech., Rev. 38, (1978/1979), pp. 298-309.

(9) Lee, Chung Chang, "Elimination of Redundant Operations for a Fast Sobel Operator." IEEE Transactions on System, Man and Cybernetics, (March/April, 1983), pp. 242-244.

(10) Goshtasby, S. H. Gage, Bartholic, J. F, "A Two-Stage Cross Correlation Approach to Template Matching." IEEE Transactions on Pattern Analysis and Machine Intelligence, (May, 1984), pp. 374-378.

VITA 2

William Glenn Patterson

Candidate for the Degree of

Master of Science

Thesis: AN APPROACH FOR MACHINE VISION RECOGNITION OF SEVERELY SKEWED CHARACTERS

Major Field: Electrical Engineering

Biographical:

Personal Data: Born in Oklahoma City, Oklahoma, April 1, 1958, the son of W. Ralph and Gloria A. Patterson. Married to Karen A. Stephens on May 19, 1984.

Education: Graduated from Memorial High School, Tulsa, Oklahoma, in May, 1976; received Bachelor of Science Degree in Electrical Engineering from Oklahoma State University in December, 1980; completed requirements for Master of Science degree at Oklahoma State University in December, 1988.

Professional Experience: Electronic Engineer, Oklahoma Air Logistics Command, Department of Defense, March, 1981, to June, 1984; Electrical Engineer, Lockheed Engineering Management and Service Co., Houston, Texas, June, 1985 to April, 1988 on the Engineering Support Contract to NASA at the Johnson Space Center; Software Engineer, Lockheed Engineering Science Co., Houston, Texas, on the Software Support Environment for the Space Station project, April, 1988 to present.