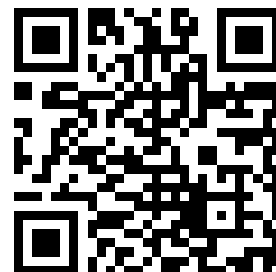

This is a reproduction of a library book that was digitized by Google as part of an ongoing effort to preserve the information in books and make it universally accessible.

GoogleTM books

<https://books.google.com>



Stanford University Libraries



3 6105 025 637 757



VISUAL IDENTIFICATION OF PEOPLE BY COMPUTER

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON THE GRADUATE DIVISION
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By

Michael David Kelly

August 1970 ^{1R}

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Buddy

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

John M. Carthy

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

W. F. Miller

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Peter C. Hart
(Stanford Research Institute)

Approved for the University Committee
on the Graduate Division:

Lincoln E. Moses

Dean of the Graduate Division

ACKNOWLEDGEMENTS

I would like to express my appreciation to my thesis advisor, Professor D. R. Reddy, for his guidance, advice, and encouragement. I am grateful to Dr. Peter E. Hart for his helpful suggestions and to Dr. Hart and Professors John McCarthy and William F. Miller for their constructive readings of this thesis.

I am also indebted to R. Gary Goodman and J.M. Tenenbaum for many enlightening discussions, to Gilbert Falk for the use of Figures 3-1 and 3-2, to Karl Pingle and Richard Neely for making available to me some of their programs, and to Joan Kelly for her outstanding typing and drawing and for her constant patient encouragement.

I dedicate this work to my wife Joan.

This research was supported in part by the Advanced Research Projects Agency of the Office of the Secretary of Defense (SD-183).

TABLE OF CONTENTS

Chapter	Page
1. INTRODUCTION ,	1
HISTORY ,	4
ORGANIZATION ,	5
2. THE PROBLEM - RECOGNITION OF PEOPLE ,	6
GOALS ,	6
OUTLINE OF THE METHOD ,	8
PREVIOUS WORK ,	12
3. PICTURE ANALYSIS AND DESCRIPTION BY COMPUTER ,	15
APPLICATIONS ,	15
EDGE DETECTION ,	20
4. GOAL-DIRECTED PICTURE PROCESSING ,	28
TOP-DOWN ANALYSIS ,	28
MODELS ,	35
5. PICTURE PROCESSING USING "PLANNING" ,	37
AN EXAMPLE: EDGE DETECTION USING PLANNING ,	40
6. LOCATING AND MEASURING THE BODY ,	46
EQUIPMENT, PROGRAMS, AND INPUT SPECIFICATIONS ,	46
CHARACTERISTICS OF THE PICTURE ,	47
APPROXIMATE LOCATION OF THE BODY ,	49
MEASURING HEIGHT ,	56
MEASURING WIDTHS ,	60

CONTENTS (continued)

Chapter	Page
7, FINDING THE OUTLINE OF THE HEAD	65
REDUCE IN SIZE	67
FIND ALL EDGES	67
FIND HEAD IN REDUCED PICTURE	72
USE PLAN TO FIND OUTLINE IN ORIGINAL	73
SPEED COMPARISON	76
LOCATE MEASUREMENT POINTS	79
8, FINDING THE FEATURES OF THE FACE	81
EYES	81
NOSE	99
MOUTH	105
9, PATTERN CLASSIFICATION	108
10, RESULTS AND CONCLUSIONS	113
FUTURE WORK	157
References	161
Appendix A TWO PICTURES OF BODIES: INPUT DATA .	167
Appendix B TWO PICTURES OF HEADS: INPUT DATA . .	180
Appendix C PROGRAMS FOR LOCATING THE OUTLINE OF THE HEAD	197

LIST OF TABLES

Table Number		Page
6-1	Standard positions for width measurements .	62
7-1	Planning speed comparison	77
10-1	Coordinate measurements from 4 body pictures	124-125
10-2	Measurements for identification test	152-153
10-3	Normalized measurements	154-155
10-4	Weights	156
10-5	Recognition distances	158

LIST OF ILLUSTRATIONS

Figure Number		Page
2-1	Measurements from the body	10
2-2	Measurements from the head	11
3-1	Grey scale picture of blocks	21
3-2	Edges from Figure 3-1	21
3-3	Notation for points used in describing gradient operators	23
4-1	Logical flow of a bottom-up picture analyzer	30
6-1	Input photograph, Subject in standard recognition position	48
6-2	Binary valued, smoothed, reduced size picture used for locating the body	48
6-3	Flow chart - approximate location of the body	50
6-4	Procedure CONNECTED	54-55
6-5	Ideal top of head	57
6-6	Template for top of head	57
6-7	Actual top of head	57
6-8	Measurements of width	61
6-9	Schematic diagram of placement of width measurements	61
6-10	Allowance for the variance from vertical	63
7-1	Unprocessed input picture of a man's head	66
7-2	The outline of a head, the result of processing Figure 7-1	66
7-3	Unprocessed input picture	68
7-4	Results of a gradient operator	68
7-5	Reducing size by averaging	68

ILLUSTRATIONS (CONTINUED)

Figure Number		Page
7-6	Results of edge-finding operator	69
7-7	The "plan"	69
7-8	The outline of the head	69
7-9	Operation of plan follower	75
8-1	Left-handed coordinate system used	82
8-2	Horizontal cross-section which characterizes the eye	84
8-3	The model for eye location on an average head	86
8-4	Constants used in predicting the location of the eyes	86
8-5	Predicting the location of the eyes on a given head	87
8-6	Procedure for finding the dark area around the eye	89
8-7	Dynamic threshold setting	93
8-8	The dark region around the eye and characterizing parameters	94
8-9	The area searched for the iris	94
8-10	Nose region in unprocessed picture	100
8-11	Essentials of the model of the eye-nose relationship	100
8-12	Locating the position of the nostrils	103
10-1	Picture KELB1	115-116
10-2	Picture KELB4	117-118
10-3	Picture JZCB1	119-120
10-4	Picture JZCB2	121-122
		117-118
	viii	119-120
		121-122

ILLUSTRATIONS (CONTINUED)

Figure Number		Page
10-5	Explanation of measurements in Figures 10-6 through 10-29	126
10-6 through 10-29	Pictures of heads with resulting outlines and features through	127 150

CHAPTER 1

INTRODUCTION

Most present day picture processing programs are unsuitable for the analysis of naturally occurring visual scenes. When presented with objects such as trees, cars, and people, these programs would become hopelessly confused. What is more, after looking at the structure of these programs, it becomes obvious that there are no simple generalizations of these techniques which will succeed. This thesis describes an attempt to develop a computer program which performs a complex picture processing task. The task is to choose, from a collection of pictures of people, those pictures that depict the same person.

In brief, the program works by finding the location of features such as eyes, nose, or shoulders in the pictures. Individuals are classified by measurements between such features. The interesting and difficult part of the work reported in this thesis is the detection of these features in digital pictures.

The methods of the program as well as the goals of the research are summarized in the next chapter. In general, the primary purpose of this research has been directed toward the development of new techniques for picture processing. The identification of people is only an interesting problem that can be used for this purpose. It

is an interesting problem because pictures of people are so complex, and there is great variation between pictures. This is in sharp contrast to the simple geometrical solids used so often in research in picture processing.

No one has previously dealt successfully with complex objects such as people in computer picture processing. The success of the program is due to and illustrates the heuristic use of context and structure. A new, widely useful, technique called planning has been applied to picture processing. Planning is a term which is drawn from artificial intelligence research in problem solving.

The bulk of the work of this thesis has been devoted to methods of finding low level features such as eyes or shoulders in digital pictures. Many of the methods developed by previous researchers have been applied to different parts of the problem. The large number of methods that are used can be indicated by summarizing the techniques used in recognizing each part of the picture. Each method is discussed more fully in the remainder of the thesis. The position of the body is located by subtraction of the background. The top of the head and the feet are identified by template matching. The edges of the head, neck, shoulders, and hips are found by edge detection operators. The outline of the head is obtained using planning to guide goal directed search after using edge detectors. The eyes are found by dynamic threshold setting followed by smoothing.

and template matching. The nose is located by dynamic threshold setting. The mouth is found with a line detection operator. All of these methods are applied heuristically in a goal directed manner which is based on an implicit model of the structure of the human body.

A large part of the success of this research is due to the availability of powerful interactive computer facilities for experimentation with the picture processing algorithms. Most often in devising algorithms for identifying particular features, it was found that intuition was faulty. The experimental procedure used to develop the final methods went somewhat as follows. For a particular feature a method for identification would be postulated. This method would be programmed and tried. Usually the method would fail. Using the display terminals available on the time-sharing system it was possible to watch the progress of the algorithm while it was running. Parameters could be varied interactively. This led to insight as to why the method failed and would suggest new methods or modifications. This "postulate-try-fail" loop usually had to be repeated many times before a satisfactory method was found. Once the method was found, "fine-tuning" was made feasible by the interactive facilities.

HISTORY.

My first interest in this problem began in a class project in the winter of 1967. A small amount of effort was devoted to this problem until the spring of 1968 when serious work began. The past two years have been devoted to this thesis research and to a study of basic picture processing procedures.

At first one of the goals was to recognize people on-line as they were standing in front of the TV camera. Eventually it turned out that the characteristics of the TV input system available prevented this. The chief problem is the inability of the system to provide detail over a wide range of light intensity values. If the entire video signal is digitized, very little detail is present. In order to get facial details, the video signal must be clipped top and bottom and the resulting narrow window digitized. This leads, however, to the necessity of adjusting external lighting, TV sensitivity, and clip levels for each person. This is time consuming and so error-prone as to be impractical. It demands too much of the patience of a volunteer subject. The work of J. M. Tenenbaum [1970], on automatic accommodation of visual parameters, promises improvement in this area.

As a consequence of the limitations on the quality of TV input, the following scheme was adopted for obtaining acceptable pictures. A number of pictures of people would

be taken in a picture taking session. Later these pictures would be examined visually by printing them on a line-printer using an approximate grey-scale (about 5 minutes per picture) or displaying them on a storage tube (about 20 minutes per picture). Those pictures which contained sufficient detail would be retained for processing. Minor faults did not cause a picture to be excluded; this was not a selection to fit the peculiarities of the recognition program.

ORGANIZATION.

This thesis is structured to discuss first the problem, next the solutions, and then the results. The next chapter will outline the goals of this research and relevant past work. Chapters 3 to 5 discuss the solutions in broad terms and in comparison with alternative approaches. Chapter 3 surveys picture analysis and description. Chapter 4 discusses goal-directed picture processing and the use of models. Chapter 5 covers the new technique, planning. Chapters 6, 7, and 8 examine the recognition algorithms in detail. Chapter 9 outlines the use of the nearest neighbor classification algorithm. The final chapter summarizes results and offers suggestions for further work.

CHAPTER 2

THE PROBLEM - RECOGNITION OF PEOPLE

This chapter will describe the overall goals and methods of the research described in this thesis. The method used for recognition will be outlined briefly to provide an overview of the structure of the program. In addition, past work on the recognition of people by computer will be summarized.

GOALS.

There is great potential for enhancing the usefulness of computers by the addition of visual input capabilities. The primary goal of the research reported here has been the development and improvement of techniques for computer picture processing.

The term "picture processing" is used in this thesis to mean the processing of a picture obtained from the outside world. It includes areas often called "pictorial pattern recognition" and "picture analysis and description". It excludes generation of pictures by computer, so-called "computer graphics". This corresponds to Rosenfeld's usage of "picture processing" [1969a], (Miller and Shaw [1968] ascribe a wider meaning to the term.)

An effort has been made to apply heuristic methods, drawn from artificial intelligence research in problem

solving, to computer picture processing, "A heuristic is a rule of thumb, strategy, trick, simplification, or any other kind of device which drastically limits search for solutions in large problem spaces" (Feigenbaum and Feldman [1963], p. 6). The term artificial intelligence has been used with a wide variety of meanings by specialists in various fields. In this thesis artificial intelligence research is considered to be complex problem solving by computer (of problems which may require intelligence for their solution) using heuristic techniques. Examples of such research include the General Problem Solver of Newell, Shaw, and Simon [1959]; programs that play checkers and chess (Samuel [1967], Greenblatt et al. [1967]); and the Dendral program of Lederberg and Feigenbaum [1968] for inference in organic chemistry.

Effective computer picture processing will probably come about via the incorporation of heuristic methods. The problems to be solved are large, complex, and not well understood. Adaptation to picture processing of generally effective heuristics used by artificial intelligence workers appears to be the best way to attack these problems. Surprisingly, in the past there has been relatively little interaction between these two research areas.

Advancement of computer picture processing can come from work on a specific problem. Past experience is meager and generally useful concepts are few. The primitive state

of knowledge makes it appropriate to attempt a particular problem with hopes of generalizing the results. This is what has been done in the work reported here.

The problem which has been chosen is the following: develop a program which will identify people from pictures taken by a TV camera attached to a computer.

OUTLINE OF THE METHOD.

The general scheme of operation of the program can be summarized as follows. Two pictures of the individual to be recognized are read into the computer. One is a picture of the entire body, head to feet. The other is a close-up of the head. The program processes the pictures to locate feature points such as the irises of the eyes, nostrils, the top of the head. Once these points are found, measurements are derived from them such as height, distance between eyes, width of head, etc. These measurements are then used in a pattern classification algorithm to extract the identity of the person from a dictionary containing known individuals and their measurements.

The method outlined above appears quite straightforward: locate features, obtain measurements, classify pattern. Obviously, obtaining measurements once feature points are located is a trivial operation. The final stage of the method, pattern classification, has been well studied. Given a good set of features, there are standard

classification algorithms which may be used. However, the first step, locating the features in a digital picture, is a process about which very little is known.

This then is the main effort of this thesis research: accurately locating desired specific points on pictures of people. Such low level picture processing, called variously feature extraction, pre-processing, or characterization of the picture, is generally recognized as the most difficult part and the principal problem in pattern recognition. (See, for example, Ho and Agrawala [1968], p. 2102.) It is worthwhile emphasizing, in view of the fact that so much previous work in pattern recognition has been concerned with mathematical techniques for classification, that in this work classification has received only minor attention.

Measurements from the face and body should provide a good means of identifying people. The reason for this expectation is that physical measurements were the basis of the Bertillon system for identification of people, which had wide use in police work prior to the discovery of the usefulness of fingerprints (Thorwald [1965]). This expectation is confirmed by the results obtained by Bledsoe which are described later in this chapter.

The measurements which have been selected for use in this thesis are given in Figures 2-1 and 2-2. Each measurement is normalized by dividing it by the measured

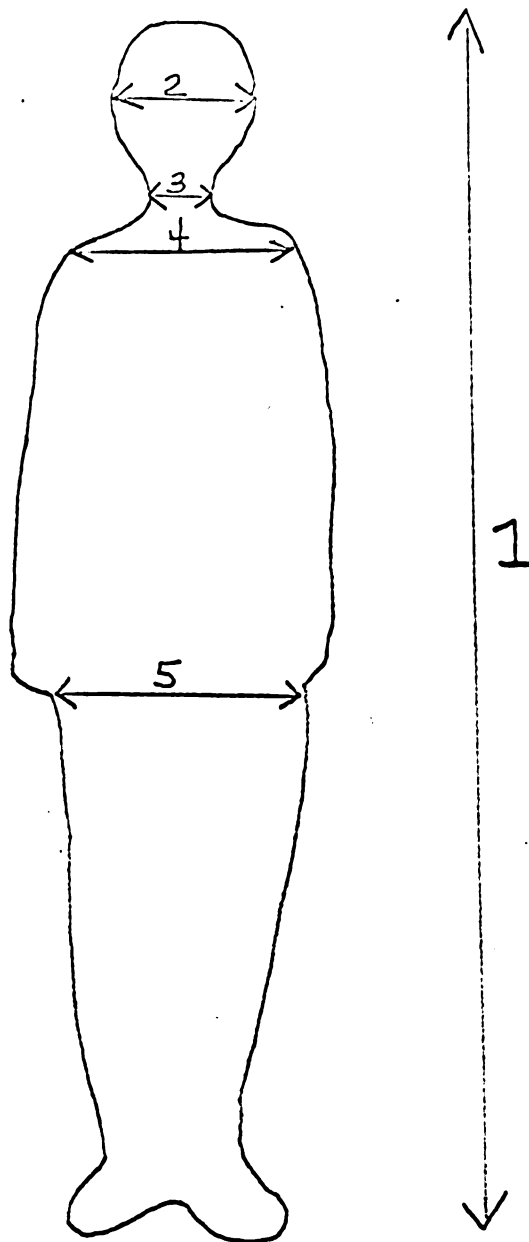


Figure 2-1. Measurements from the body.

1. Height.
2. Width of head.
3. Width of neck.
4. Width of shoulders.
5. Width of hips.

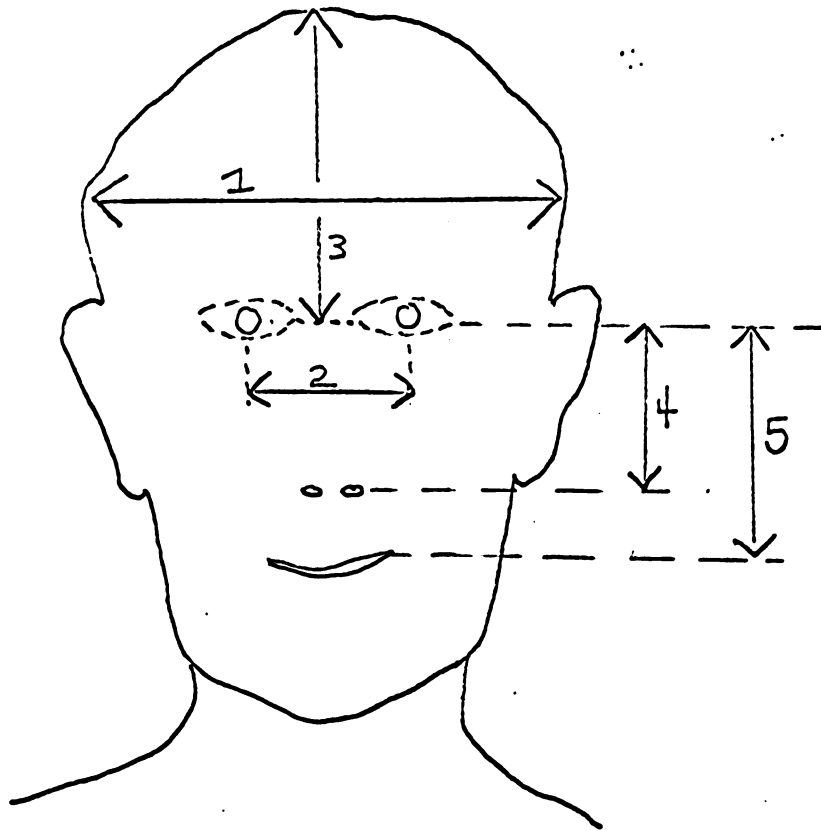


Figure 2-2. Measurements from the head.

- 1, Width of the head,
- 2, Distance between eyes,
- 3, Distance from top of head to eyes,
- 4, Distance from eyes to nose,
- 5, Distance from eyes to mouth,

height.

PREVIOUS WORK,

A number of papers have appeared in which pictures of faces have been partially processed by computer and the results displayed. Examples are the papers by Narasimhan and Forango [1964] and Hueckel [1969]. Both of these papers present computer produced line drawings of faces which were derived from grey scale input pictures. Such operations are fine for presenting faces to the human eye but represent only a very small step toward computer description and recognition.

The principal prior work on recognition of people by computer has been done by W.W. Bledsoe. This work was begun at Panoramic Research, Inc. and continued with P.E. Hart at Stanford Research Institute. (See Bledsoe [1964] and [1966]).

There are differences and similarities between the work of Bledsoe and the work reported here. The chief difference is that Bledsoe created a man-machine system in which a human operator, working with a face projected on a Grafacón or "Rand tablet", located the feature points on the face and manually pointed out their position for the computer to record. In contrast, my work consists primarily of an attempt to automate this feature location step. Bledsoe was concerned with recognition of photographs of faces; I

consider both body and face, Biedsoe permitted a wide variation in head rotation, tilt, lean, photograph quality, and light contrast; I require much more standardization of pose and can obtain it since I control the picture taking environment. In spite of these differences, the work reported here follows the basic idea for visual identification of people first laid out by Biedsoe: find the measurements and use them for identification. Another way of summarizing this is: automate the identification techniques of Bertillon.

Biedsoe's results verified that facial measurements made on photographs could be used effectively for facial recognition. In his work, measurements were obtained from 2000 photographs, 2 photographs for each person in the sample. Given a set of measurements for an unknown person, the classification system attempted to supply a name or a small list of names which included the unknown individual. Using various classification methods Biedsoe found that an average reduction in uncertainty of about $1/100$ could be obtained.

Biedsoe's group was also concerned to a limited extent with finding features automatically (Blisson [1965a], [1965b]). The results of this were inconclusive; many problems were encountered trying to determine the location of feature points. Biedsoe's success with facial classification using measurements while leaving open the

problem of automatic feature location has been a stimulus for the work reported in this thesis.

Sakai, Nagao, and Fujibayashi [1969] have reported their work on finding faces in photographs. Their goal is to detect if a face or faces are present in a picture. They first produce a picture which contains the edges of the input picture. A large oval template corresponding to the head outline is then matched with the edge picture. All reasonable positions and sizes of the template are tried. In those positions where the oval template receives a high response, the head hypothesis is checked by further template matching that expects many edges in the eyes, nose, and mouth and few edges on the forehead. This method appears to be time consuming, and the result is only an approximate location for the head in the picture.

Three Russians, El'bur [1967], Yurans [1967], and Rastrigan [1967], have presented methods for identifying faces from photographs. The methods assume that a representation of a face as a set of points is available. The papers are mostly on projective geometry; there is very little mention of application. Hart [1969] has prepared a summary of the content of these papers.

CHAPTER 3

PICTURE ANALYSIS AND DESCRIPTION BY COMPUTER

This chapter will discuss past work in automatic picture analysis and description. Applications will be summarized briefly. Useful techniques which have been developed will then be described in some detail.

APPLICATIONS.

Applications of computer picture processing can be summarized as an area of much past work with little solid success. Many experiments have been performed which have used computers to process and identify visual images. In only one area has this work passed from the experimental to the practical. This area is optical character recognition (OCR). The methods used for OCR will be discussed. Following this some of the other application areas will be surveyed.

Character reading machines are a practical success. To quote a long time leader in the development of the field, "It is now possible to read with any desired accuracy any reasonably good printing whether done by typewriter, high-speed printer, or typesetter" (Rabinow [1968], p.24). In many ways, the problems encountered in reading characters are different from and simpler than the problems found in more general picture processing. Nevertheless, because of

the great success which has been achieved by OCR, it is appropriate to examine the methods which have been used to achieve this success so that their potential for other types of picture processing may be considered.

One method which has been used for successful OCR is special input, designed for recognition. The many character sets intended for machine reading are examples of this. Rabinow reports building a high resolution machine to read alphanumeric characters specifically designed for machine reading. "It read without error, that is, it has read several billion characters without a single error and the reject rate is something of the order of one character in 2 million" ([1968], p.7). Control of the input with recognition in mind must be considered in any general picture processing task.

Special character sets are not the only characters which can be read by reading machines. In general however it is the fact that printed characters are black against a contrasting background, with more or less sharp edges, that permits the use of straightforward recognition methods with success. A good discussion of the recognition methods used, the subject of the next several paragraphs, may be found in Character Recognition 1967 (British Computer Society [1967]),

Character readers generally recognize characters by template matching. The character "A" for instance is

compared with idealized specimens of "A", "B", ... , "Z" to determine the best match. The matching is not however a straightforward cross-correlation, which ignores the fact that many characters have common areas and that some areas are nearly unique. Rather, a weighted mask is correlated with the unknown character. In particular, the unknown character is usually represented as grey values on a rectangular matrix A_{ij} . For each character, C , to be recognized there is a weight matrix W_{ij}^C . The score for character C is then $S^C = \sum W_{ij}^C A_{ij}$ (ignoring character positioning). The weights for particular characters are most often obtained heuristically and intuitively. This sort of weighted template matching can be useful in picture processing when the object to be found can be reliably characterized as to shape and relative grey values.

Another recognition method used in OCR is stroke analysis. In this method a character is classified by a group of characteristic properties and features. For instance a "T" might be required to consist of a long vertical stroke with a horizontal stroke at the top. The individual strokes may be found using weighted templates as before. This method is currently used on highly stylized characters. However, it is an example of the sort of two-stage method that is necessary when the input has wide variability.

In contrast to the success of optical character

recognition, applications in other areas are only partially successful or are experimental.

Recognition of blocks and simple geometrical solids has been the goal of much interesting recent research. The first work of this nature was done by Roberts in his work "Machine perception of three-dimensional solids" [1963]. This was a computer program which processed and recognized pictures of blocks and wedges. From a photograph, a line drawing of the scene was extracted. The line drawing was processed and a list of the three-dimensional objects in the scene was produced. Once the object list was obtained, various two-dimensional projections of the objects could be displayed. This work was particularly noteworthy in two respects. First, solutions were developed for the difficult problems encountered in going from the representation of a picture as a matrix of light intensities to the representation of a picture as a line drawing. Roberts' edge detection operator will be discussed in detail later in this chapter. Secondly, Roberts introduced the use of a four-dimensional, homogeneous coordinate system to handle perspective transformations.

The Stanford University Artificial Intelligence Project is attempting to develop visual and motor capabilities for computers. Building on the work of Roberts, several programs have been written which can recognize blocks on a tabletop. Work is currently in progress directed toward

recognizing more complicated geometrical solids, processing pictures of roads for a computer controlled car, accommodation of vision system parameters for enhanced visual perception, using texture to distinguish areas of interest in pictures, and developing models and data structures for the representation of scenes. (See McCarthy et al, [1968], Feldman et al, [1969], Falk [1969], and Paul et al, [1969])

At Stanford Research Institute a computer controlled robot is being developed (Raphael [1968], Nilsson [1969]). Vision programs are being supplied which can cope with the robot's environment, a room with large blocks, wedges, and platforms.

Guzman [1968], working at the M.I.T. Artificial Intelligence Project, has developed a program which decomposes a list of lines into a list of objects. Guzman's program can handle scenes of far greater complexity than the programs mentioned above. However, the input to this program is a symbolic and error-free list of the edges in the scene.

From the preceding paragraphs it may be seen that there is much work directed toward computer recognition of blocks and solids. In examining this work, however, the fact stands out that even for simple scenes the computer processing of visual images is difficult and not clearly understood.

Other application areas where computer picture processing techniques have been used with some success include the following of particle tracks in bubble, cloud, and spark chamber photographs; the processing of photomicrographs, particularly those of chromosomes; the processing of aerial photographs to obtain information on cloud types and terrain; and the visual processing of fingerprints.

EDGE DETECTION.

In picture processing one of the most important problems is edge detection.

Much work on edge detection in digital pictures has been reported. An "edge" is the boundary between two objects or between an object and the background. It is contrasted with "line" which denotes a thin stroke against a uniform background. (The lines in Figure 3-2 represents edges in Figure 3-1.)

Finding the edges in a picture is important, if the picture is to be analyzed and described by a computer. Much of the important information in a picture is contained in the edges. This may be seen in Figure 7-4 which shows the edges present in Figure 7-3. It is evident that most of the information of Figure 7-3 is retained.

Edge detection has received considerable attention as a part of computer picture processing. Roberts [1963], in

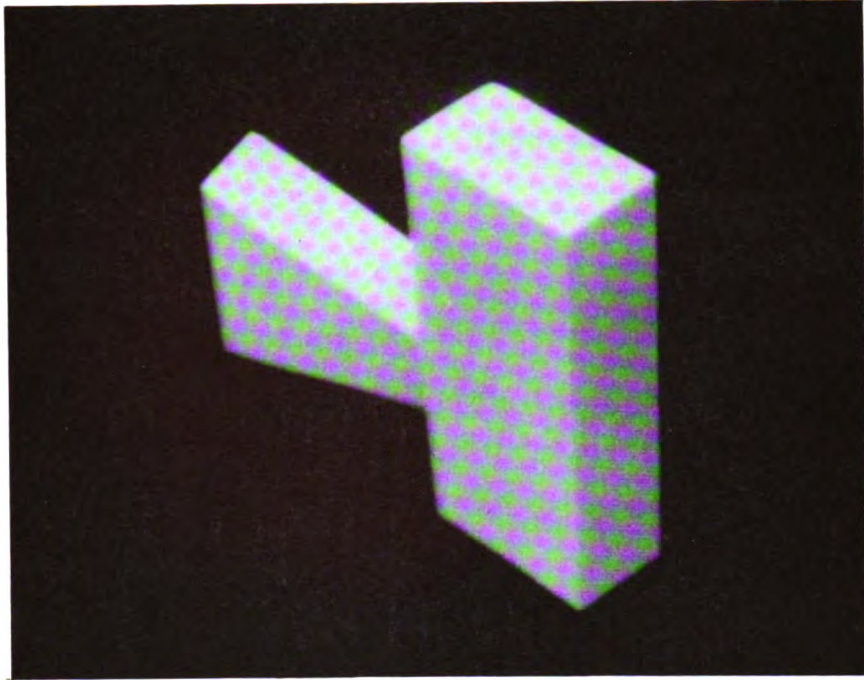


Figure 3-1. Grey scale picture of blocks.



Figure 3-2. Edges from Figure 3-1.

his pioneering work in machine perception, detected edges in the picture as the first step of processing. Narasimhan and Fornango [1964] emphasized the importance of edges in experiments with pictures of human faces. Guzman [1968], in his work on the analysis and description of scenes, assumed that all edges in the picture had been found accurately. Other research which illustrates the importance of edges in pictures includes the Stanford University Artificial Intelligence Project (Feldman et al. [1969]), the S.R.I. Robot project (Forsen [1968]), and Rosenfeld et al. [1969b].

Edge detection in pictures may be considered a high pass filtering operation. If the high spatial frequencies in a picture are emphasized, and the low spatial frequencies are suppressed, the result approximates a line drawing of the scene. This technique has been used to enhance contrast in lunar photographs (Billingsley [1967]).

Edges in digital pictures are usually detected, however, by the use of local operators. Such operators examine and compare intensity values within a small region of the picture. Most often this operator is some variant of the gradient, the derivative in the direction of the maximum change of intensity. Figure 7-4 is an example of the results of the application of a gradient operator to the picture of Figure 7-3. Mathematically the gradient of a function $f(x,y)$ of two variables is the vector given by

$$\overrightarrow{\text{grad}(f)} = \frac{\partial f}{\partial x} \vec{i} + \frac{\partial f}{\partial y} \vec{j}$$

where $\vec{i}$ and $\vec{j}$ are unit vectors along the x and y axes. Many different approximations to the gradient have been used. Some of these are discussed below.

$Z_{i,j}$	$Z_{i+1,j}$	$Z_{i+2,j}$	A	B	C
$Z_{i,j+1}$	$Z_{i+1,j+1}$	$Z_{i+2,j+1}$	D	E	F
$Z_{i,j+2}$	$Z_{i+1,j+2}$	$Z_{i+2,j+2}$	G	H	I

Figure 3-3, Notation for points used
in describing gradient approximations.

Roberts [1963] used the following discrete approximation to the gradient. Let Z be the picture matrix and denote the intensity value at elements $Z_{i,j}$, $Z_{i+1,j}$, ..., $Z_{i+2,j+2}$ with the letters A, B, ..., I as in Figure 3-3. Then the magnitude of the gradient at point $Z_{i,j}$ is given by

$$|\vec{G}| = \sqrt{(E - A)^2 + (B - D)^2}.$$

Robert's gradient operator is quite sensitive to noise. To reduce this problem Sobel and Feldman developed a gradient operator which is described by Pingle [1969]. Their approach was to consider the nine points in a 3×3 square. The difference between the intensity at each outer point and the center is weighted by an appropriate vector. Using the notation of Figure 3-3, the gradient at point

$z_{i+1,j+1}$ is given by

$$\begin{aligned}\vec{G} = & (A-E)[-1,1] + (B-E)[0,2] + (C-E)[1,1] \\ & + (D-E)[-2,0] + (F-E)[2,0] \\ & + (G-E)[-1,-1] + (H-E)[0,-2] + (I-E)[1,-1].\end{aligned}$$

This can be reduced to

$$\vec{G} = U[1,0] + V[0,1]$$

where

$$\frac{\partial f}{\partial x} \approx U = \frac{C + 2F + I - A - 2D - G}{8}$$

and

$$\frac{\partial f}{\partial y} \approx V = \frac{A + 2B + C - G - 2H - I}{8}.$$

It is interesting to note that the lack of sensitivity to noise of this operator is due to the fact that it is equivalent to smoothing and then differencing. Let the picture be smoothed by replacing the intensity at each point with the average intensity in a 2x2 square containing the point, e.g.,

$$A' = (A + B + D + E) / 4$$

$$B' = (B + C + E + F) / 4$$

$$D' = (D + E + G + H) / 4$$

$$E' = (E + F + H + I) / 4 \text{ etc.}$$

Now let $\frac{\partial f}{\partial x}$ be approximated simply by

$$\frac{\partial f}{\partial x} \approx U' = \frac{B' + E' - A' - D'}{2}.$$

If the definitions of the primed variables are substituted

In this formula, the result

$$\frac{\partial f}{\partial x} \approx U' = \frac{C + 2F + I - A - 2D - G}{8}$$

is exactly the same as Sobel and Feldman's approximation,

Sakai et al, [1969] also use the nine points of a 3×3 square for their approximation to the gradient. Referring again to Figure 3-3, the magnitude of the gradient at point $Z_{i,j}$ is given by

$$|\vec{G}| = E - \min(A, B, C, D, E, F, G, H, I).$$

Rosenfeld and his associates (Rosenfeld et al, [1969b], Abbamonte et al, [1970]) noticed that an edge detector which considers only a few points is sensitive to noise. On the other hand, if an edge detector considers many points over a wider area, it will smooth out noise but it detects major edges in a wide range of positions around the actual position. They suggest that the product of the output of several small and large operators will combine the advantages of both,

The output of the product operator will only be high if all of the constituent operators have high output. Thus at an edge there will be a sharp peak along with noise suppression. To implement this idea, they suggest the use of the following approximation to the gradient. "Let H_k denote the absolute difference between averages taken over two vertically adjacent, non-overlapping k -by- k squares on opposite sides of a given point" [1970, p. 17]. Then at

that point

$$\frac{\partial f}{\partial y} \approx \prod H_{k_i} \quad k_i = 1, 2, 4, 8, 16$$

and

$$|\vec{G}| \approx \max\left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right).$$

Hueckel [1969] has developed an elaborate edge detection operator which considers all points in small circles of the picture. These circles contain from 32 to 177 picture points. Within these circles a set of functions $F(x,y)$ representing ideal edges are defined. The intensities found in an actual circle represent a function $E(x,y)$. Finding the edge in a circle then becomes the problem of finding the particular function $F(x,y)$ which minimizes the distance

$$L = [E(x,y) - F(x,y)]^2.$$

The size of L is a measure of how edge-like the edge is. By constraining the problem suitably, Hueckel has found a straight-forward algorithm for implementing this minimization. Hueckel's operator gives good results on circles which contain one edge or at most two edges. His algorithm is, of course, much slower than the operators mentioned earlier that use fewer points.

In addition to the gradient, it has been suggested that higher derivatives (Williams [1965]) or the Laplacian (Rosenfeld [1969a], Bell [1968]) should be used for finding edges. These methods seem unsatisfactory because of their tendency to amplify noise.

Edge followers are used to combine the edge information produced by a local operator into a line that represents the edge. An edge follower is an algorithm that searches for and follows an edge. It uses the direction and curvature of edges to connect adjacent short segments. Edge following is discussed by Rosenfeld [1969a, p. 136], Descriptions of working edge followers are given by Pingle [1966], by Pingle et al. [1968], and by Greenblatt et al. [1966]

Besides edge detection one of the most common techniques encountered in picture processing is template matching. This method was mentioned briefly in the discussion of optical character recognition. In OCR this technique has been very useful. However, in more general picture processing problems the technique usually loses its effectiveness. The reason for this is that, except for very simple objects in constrained situations, it becomes almost impossible to develop an idealized prototype.

Another picture processing technique that is generally useful is smoothing. Smoothing is the modification of the grey level at a point in such a way as to make its value more similar to the values of its neighbors. Typically this is an averaging operation. The greatest value of smoothing is that it reduces the noise that is present in almost all pictorial input. It is also helpful in reducing minor surface variations caused by texture and shadow.

CHAPTER 4

GOAL-DIRECTED PICTURE PROCESSING

TOP-DOWN ANALYSIS.

The program described in this thesis uses top-down picture analysis. It is believed that the explicit use of this approach is one of the major reasons for the success of this program.

The use of the terms "top-down" and "bottom-up" to describe picture analysis methods was originated by Shaw [1968]. The names are derived from a loose analogy which can be drawn to methods with similar names which are used for syntax-directed analysis of programming languages. Such analysis or "parsing" is based on the structure of the programming language. The structure of the program is expressed by a set of "productions" or rewriting rules. The principal methods used are discussed in a paper by Feldman and Gries [1968]. A bottom-up parse of a program starts with the characters and symbols of the program and attempts to combine them together using the productions in reverse order until the whole assemblage represents a correct program. In contrast, a top-down parse works in the opposite direction. Starting from the highest level goal (a program) a search is made among the alternative constituents of each level of production until eventually a series is found which includes the characters of the input.

In discussing analysis of pictures, "top-down" and "bottom-up" are not used to describe formal mechanisms such as are used in programming language translators. Rather, bottom-up analysis will be used to describe schemes which first process the picture exhaustively at low levels. Top-down analysis will refer to goal oriented processing which searches for the constituent parts of the objects searched for in the program.

Bottom-up analysis has been used widely in picture analysis. It consists of a series of more or less independent processing steps which are applied sequentially in an attempt to transform an input picture into a description of the picture. Each stage of processing reduces the amount of information that will be passed on to the following stage. The idea is to first concentrate on the lowest level of detail and then to consider higher levels one at a time until finally the entire picture has been analyzed and described.

The schematic diagram given in figure 4-1 is an example of the typical organization of a bottom-up processor. The first stage is called region analysis. This usually consists of low level operations such as detecting edges or homogeneous areas. The output of this first stage would be a list of all of the edges or all of the regions. The second stage is called region description. This part of the program collects subsets of its input into recognizable

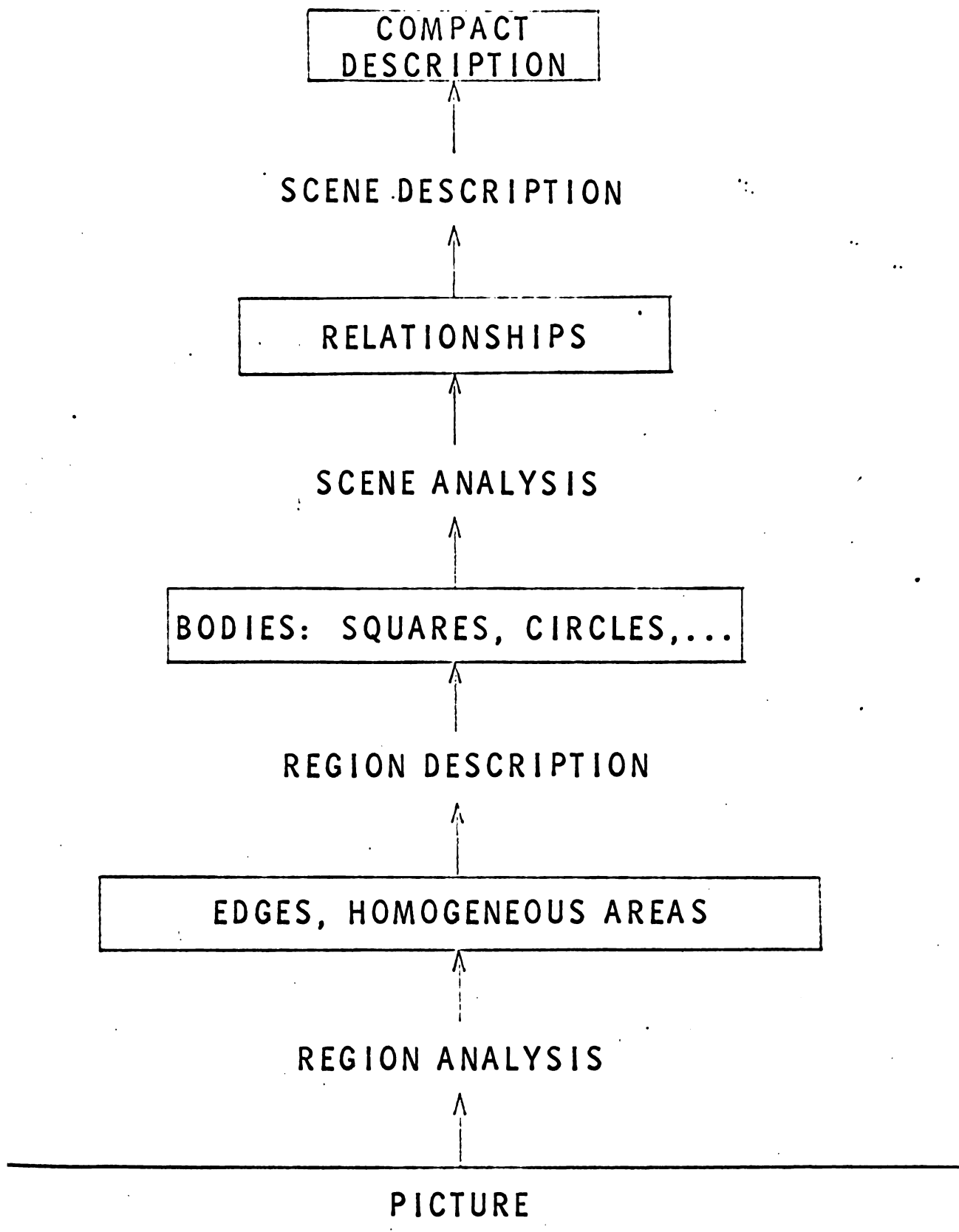


Figure 4-1. Logical flow of a bottom-up picture analyzer.

small pieces of the picture. For instance, the output of this stage might be a list of rectangles or ellipses or blobs of a certain shape along with their coordinates. The third stage is called scene analysis. In this stage relationships between regions are considered. Examples of such relationships are "next to", "above", "close to", or "within". Thus the output of this stage contains the relationships between the regions in the picture. The last stage is called scene description. Here the final description is obtained. For instance, a square adjacent to two parallelograms may represent a cube, or an oval which contains two circles, two dots, and a line may represent a face.

Most picture processing programs are organized along the lines of the bottom-up model described above (see Feldman et al. [1969]). However, for complex pictures the bottom-up method fails. The reason for this failure is that in cluttered pictures the program must know what it is looking for in order to find it. The bottom-up strategy would work if pictures were perfect and noiseless. However, real pictures contain much noise and irregularity. Accordingly, at each stage of processing the input contains false information as well as missing information.

To avoid the problems inherent in bottom-up organization, a top-down, goal directed approach should be taken. In such an approach the program knows of certain

classes of objects which may be present in a picture. The goal of the program is to try and find these objects if they are present. The constituent parts of each top level goal are known, as well as the parts of the parts down to the lowest level. The program searches for some of these basic parts, namely those that can be most reliably detected. If a particular low level part is found, then, because the structure of the object is known, some other known part should be present at an adjacent location. A specific routine can examine the area in question in detail to determine if such a part is present. In this way, for instance, the edge of a block might be detected once a corner is found, even though the edge might be too weak to be passed through the threshold of a gradient operation. As more and more adjacent constituent parts of a goal object are found, the confidence that an object has been found increases. Prediction of the location of the remaining parts of the object becomes easier and easier. Fairly straightforward methods can be used to verify whether these parts are indeed present. Thus, as recognition proceeds the amount of work to be done diminishes.

Top-down analysis is needed because it considers global information about the structures and interrelationships of the objects in the picture. It would be desirable for that part of the program which directs the search for the constituents of a goal object to be general purpose. To

such a general purpose program one could specify in some higher level notation or "language" the structure of objects. For instance, a "head" could be described as:

- round on top.
- somewhat flat on the sides.
- curving inward for the neck below the sides.
- etc.

The same program could then be used to identify cows or battleships (for instance) using a different structural description. Such high level descriptions are the goal of picture description languages.

Picture description languages have much promise as a tool for picture analysis. A number of descriptive notations and processing algorithms have been suggested for this purpose (Narasimhan [1966]; Lipkin, Watt, and Kirsch [1966]; Ledley [1965]; Miller and Shaw [1967]; Clowes [1969]; Pfaltz and Rosenfeld [1969]). Excellent discussions of these linguistic picture processing techniques can be found in the paper by Miller and Shaw [1968] and in Rosenfeld's book [1969a, Chapter 10].

In general, picture analysis using picture description languages proceeds as follows. A set P of "pieces" or "primitives" is defined; these are the basic elements from which a picture description can be built up. A set C of composition operations is defined. A set R of rules or "productions" specifies allowable combinations of elements

of P and C. Then F becomes a set of pictures that can be built up using R, P, and C. If x is an unknown picture belonging to F, then F can be recognized if we can recognize the elements of P in it and the way these primitives are combined.

This sort of analysis works for simple pictures but as the pictures become complex the method breaks down. For complex pictures a choice must be made. Very simple primitives may be chosen which are easy to recognize. In this case R, the allowable rules of combination, becomes so complex and unwieldy that recognition of these combinations becomes impossible. If instead, complex primitives are chosen, these become as difficult to recognize as the picture itself.

A possible solution to this problem is to provide hierarchies of structural descriptions. Straight-forward application of hierarchical decomposition leads back, however, to a bottom-up approach to processing where one loses sight at the lower levels of the top-level structure.

Analysis of complex pictures using picture description languages will not be possible unless powerful and systematic algorithms for processing elaborate and complicated picture descriptions can be developed.

The problems discussed above can be summarized as follows. Picture description languages are not sufficiently well developed to permit the specification of the numerous

heuristic tests and considerations of interrelated structure necessary to find objects in cluttered, grey-level pictures. Because of this a general purpose analysis program based on a picture description was not used.

The methods that are used to implement goal-oriented processing are "planning" and "models". Planning will be discussed in the next chapter. The use of models is discussed briefly below.

MODELS.

A model is a description which contains some of the structure of the thing which is represented. The idea of using models of objects to be recognized by computer seems to be implicitly present in most picture recognition programs. The idealized "A" represented by a resistor matrix in a character reader, certainly is a model of an "A". It is only in recent years, however, that it has become evident that the model should be used to direct all levels of processing, not just a top level decision. Reddy [1969] discusses the explicit use of such models in his recent paper. Feldman and his associates at the Stanford Hand-Eye project are working on developing data structures and representations for models of geometrical objects. Bledsoe [1964] suggested using models of the face as a method of attacking the problem of locating facial features. If an object has been described in a picture description language,

the description is, of course, a model of the object.

Models of the structure of human bodies and faces are used throughout the processing programs discussed in later chapters and guide these programs. The form of these models varies widely. The matrix of average light intensities models the background of pictures of people. A table of average ratios between facial features models a face. A flow chart of the subroutine which distinguishes nostrils from noise models a "reasonable" nose. Because of the wide variety of uses of and needs for these models, no standard modeling formalism was used. Nevertheless, at every stage a conscious attempt has been made to incorporate goal-oriented processing directed by a model of the object desired.

CHAPTER 5

PICTURE PROCESSING USING "PLANNING"

This chapter discusses a method for performing goal-directed picture analysis. The method is an improvement on the most commonly used procedures because it considers global information more efficiently than traditional local techniques. The method is called "planning" in this paper, a term drawn from artificial intelligence research, for the method precisely fits the description of planning given by Minsky [1961]. It is believed that planning can be useful in many projects involving computer vision. In addition, it is helpful to relate the search for objects or features in pictures to the work on reduction of a given search space in the field of artificial intelligence.

Much of picture processing can be viewed as search. The space to be searched is the matrix of light intensities which represents the picture. The goal of scene analysis and description is locating the important objects in the picture. The intensity matrix must be searched to find an eye, a chromosome, or the corner of a cube. Many of the difficulties encountered in picture analysis are due to the large size of the search space. Truly, one cannot see the forest for the trees. If a processing algorithm attempts to detect features by considering local areas, as most

algorithms do, the great detail which is present in the picture obscures the larger features. A global strategy is needed.

In the field of artificial intelligence, global search is one of the central problems. Much research has been done on developing heuristics for reducing search in such fields as problem solving and game playing. A generally useful concept which has emerged is planning, first used by Newell, Shaw, and Simon in their General Problem Solver [1959].

Planning is discussed by Minsky in his paper "Steps toward Artificial Intelligence" [1961]. Artificial intelligence is considered to be mechanization of the problem solving process. Basic techniques for problem solving include search, pattern recognition, learning, planning, and induction. The following quotations from Minsky's paper define planning.

Planning is the analysis of problem structure in the large. "For really difficult problems, ... step-by-step heuristics ... will fail, and the machine must have resources for analyzing the problem structure in the large - in short, for 'planning' " [p. 21].

"Perhaps the most straightforward concept of planning is that of using a simplified model of the problem situation. Suppose that there is available, for a given problem, some other problem of essentially the same character with less detail and

complexity. Then we could proceed first to solve the simpler problem. Suppose, also, that this is done using a second set of methods, which are also simpler, but in some correspondence with those for the original. The solution of the simpler problem can then be used as a 'plan' for the harder one." [p. 25]

The steps outlined in the preceding paragraph can be applied almost directly to the problem of finding desired objects in a picture. This represents an entirely new application for planning. It has not been used previously in picture processing.

In brief then, picture search using planning consists of three simple steps. A new digital picture is prepared from the original; the new picture is smaller and has less detail. Objects are tentatively identified in the reduced picture. The tentative analysis of the reduced picture, a list of the objects found and their locations, is used as a plan to verify the presence of edges in the original picture.

In a sense, the idea of using planning in picture analysis is not new. Kirsch et al, [1957] gave suggestions for possible use in processing pictorial information with a digital computer. One of the suggestions is that a defocused preliminary scan of a picture should be made. The averaging and size reduction in my implementation of

planning is very similar to defocusing and sampling. In spite of this early recommendation for a defocused scan, In the years since then the idea has been forgotten and Ignored.

AN EXAMPLE: EDGE DETECTION USING PLANNING,

It was pointed out in Chapter 3 that edge detection is necessary for picture analysis and description. Many methods for detecting edges by computer were presented. In spite of much effort, the accurate selection by computer of "important" edges in fairly cluttered scenes has not been particularly successful. The methods given for edge detection are fine for presenting information to the human eye (as Figure 3-4 shows), but what is desired is scene analysis and description within the computer. In reaching for this latter goal problems of noise, disappearing edges, and false edges intrude. Brief comments about the limitations of these methods encountered at the Stanford Artificial Intelligence Project have appeared (Pingle et al. [1968], McCarthy et al. [1968], Paul et al. [1969]). Nilsson [1969] states the problems with local methods of edge detection from experience at the S.R.I. Robot project:

"The line drawing often contains flaws that seriously complicate its analysis. Some of these flaws could be corrected by more elaborate local processing. However, there is a limit to how well

local processing can perform, and when significant edges cannot be told from insignificant edges on the basis of local criteria, the goal of producing a perfect line drawing in this way must be abandoned," [p. 513]

Global information about the edges in the picture and the structure of the objects they represent is needed. This is the idea behind the application of linguistic techniques which was mentioned earlier. Another approach to incorporating global structure into edge detection is the decision tree in use at S.R.I. (Nilsson [1969]).

The global methods for edge detection also encounter problems. Here the program looks for specific edges in specific relationships. The search space is very large, tens of thousands of points in a picture, and the combinatorics of the problem force unacceptably long search times. In the picture there are just too many lines and there is too much looking to do.

The difficult problem of locating important edges in pictures may be approached by using the three steps of planning.

- 1, Extract a simplified problem from the given problem. To simplify the problem, a much smaller picture is prepared with the intensity at each point equal to the average intensity over an area of the original picture. The new problem: find the important edges in the new small

picture.

The new problem is much easier than the original problem. Many of the small features in the original picture are no longer present. The picture has been greatly reduced in size. Only the important features of the picture remain. Less detail means fewer edges, a smaller search space, and much faster processing. Because of the smoothing done while averaging intensities, the small picture will contain little noise. Faint and blurred edges of large objects will be enhanced.

2. Solve the simpler problem. The techniques for finding edges discussed in Chapter 3 can be applied to find the desired edges in the small picture. Since only the principal objects from the original picture now remain, only major edges will be detected. As short edges are collected to form shapes they may be tested as to their acceptability. This testing should incorporate knowledge of acceptable shapes for the objects to be recognized. Since there are few edges to consider, false paths are found relatively quickly. Backtracking is far less of a problem since the data structures which must be erased are much smaller.

3. Use the solution to the simpler problem as a guide (a plan) for solving the actual problem. Within the small picture, certain edges have been found. Now it is a fairly simple matter to return to the larger picture and find the desired edges accurately. For example, a straight line may

connect points P' and Q' in the small picture, P and Q are the corresponding points in the large picture, Therefore we know that in the large picture there is an approximately straight edge which runs from the vicinity of point P to the vicinity of point Q . Since the approximate location and direction of this edge is known, it is possible to detect it quite easily and accurately. The search for this edge can be confined to a narrow band between P and Q . It will be easy to detect a false path for it will soon diverge from this narrow band.

The remainder of this section will discuss briefly some observations about edge detection using planning.

Is "planning" merely the application of a large edge detection operator? Certainly the search for edges made in the small picture could be a search of the large picture using an operator which examines a 24×24 point square. In fact, the two steps, large picture to small picture and small picture to small edge matrix, could be one step, the application of a 24×24 point operator. Alternatively, the entire large picture could be heavily smoothed and the search for edges could be done in this smoothed picture. Planning, however, has two advantages over either of the approaches listed above. First, it is much easier to design and debug the program which searches for significant edges when using planning. This is because the reduced picture is so much smaller. The use of planning aids insight. While

designing a program to find the edges in a picture of 625 points (for instance), the designer can comprehend easily both the overall structure and the detailed structure of the picture. It is very difficult to obtain the same level of understanding when the picture contains 40,000 points. The second advantage of planning is speed. Planning is 40 to 80 times faster than the same search without planning. (This factor was obtained during tests described in detail in Chapter 7.) This increase of speed is particularly important when designing search programs on an interactive computer system.

If a program incorporated more elaborate structural knowledge of the object to be recognized, could planning in a picture of reduced size be eliminated? No, such a program must be less effective for the following reasons. The combinatorics of the large search space will require excessive time. The effect of noise and irregularities in the picture will be even worse. This is emphasized by the elaborate provisions for search that had to be built into the plan follower. Even when the direction and approximate location of an edge is known, its detection can be hard. For such edges, planning is essential.

An improvement on the planning technique presented in this paper would be recursive application of planning at varying size reductions. For instance, the original picture could be reduced in size twice, four times, and eight times.

The plan found in the $1/8$ size picture could be used to find edges in the $1/4$ size picture. This could then be used as a plan to find edges in the $1/2$ size picture. Finally the accurate true edge would be found in the original picture. A hint of such a method was mentioned in the description of the plan follower. The local edge finding operator examined only alternate points. It could be considered to be using a $1/2$ size picture. The program could decide itself how much size reduction and averaging to do. At each level, if there is too much detail, a smaller picture could be called for.

CHAPTER 6

LOCATING AND MEASURING THE BODY

This chapter describes the experimental environment in which this work was done, and explains the methods used to locate measurement points in a full length picture of the person to be identified.

EQUIPMENT, PROGRAMS, AND INPUT SPECIFICATIONS,

This work was performed using the facilities of the Stanford Artificial Intelligence Project. The system consists of two computers, a PDP-10 and a PDP-6, connected as dual-processors sharing 130,000 words of core memory; a very fast, head-per-track Librascope disk for swapping; and an IBM 2314 disk storage unit for program and data storage. Several types of display consoles as well as teletypes are available for use as terminals. A more detailed description of the computer system can be found in McCarthy et al. [1968] All of the programs are written to run under the Stanford time-sharing monitor (Moorer [1969]).

Picture input is obtained from a standard vidicon TV camera. The video signal is digitized by a analog-to-digital converter and sent to memory via a high speed data channel. During one TV reading operation up to 333 samples can be taken from each video scan line and up to 256 of the alternating scan lines can be read. Thus the

maximum size picture which can be read is 256 x 333 points. Such a picture covers the entire field of view of the TV camera. Each point consists of a four bit light intensity value so that sixteen levels of grey are available. Zero indicates the darkest points; fifteen, the brightest.

The program is written almost entirely in Fortran. A few subroutines are coded in assembly language, such as those that handle input-output, list processing functions, dynamic storage allocation for picture buffers, and partial word operations. The programs occupy about 30,000 words of storage. Data storage, assigned dynamically, typically requires another 20,000 words, primarily for buffers for processed and unprocessed pictures.

CHARACTERISTICS OF THE PICTURE.

The person to be recognized stands in a standardized position against a normal room background. The person is told, "Stand in a relaxed position of 'attention', feet together, arms at your sides. Look directly at the TV camera." Figure 6-1, an input photograph of a person to be recognized, gives an example. Minor variations in position are permitted. It would, of course, be impossible to prevent such variations.

The background may vary. It normally consists of the objects in the computer room where these experiments were performed. There are several reasons for not requiring a

Figure 6-1.
Input photograph.
Subject in standard
recognition position.



Figure 6-2.
S'. Binary valued,
smoothed, reduced size
picture used for
locating the body.



standard background. When this work was begun a door-size standard background was built. However, it was still difficult to find the outline of the person. Shadows from the individual being recognized and non-uniform lighting on the background presented new problems. The solution developed for these problems permits a wider variety of backgrounds. Since the more general case is easy to handle, it appeared worthwhile to do so. Another factor which helps in permitting various backgrounds is the small depth-of-field which is characteristic of the TV lenses which were used. Objects that are very far behind the person are out of focus, hence blurred and washed out.

APPROXIMATE LOCATION OF THE BODY.

The first step in processing is to determine the approximate location of the body. A flow chart of this process is given in Figure 6-3. Each of the steps is described in more detail below.

Before beginning identification, a picture M (model) is taken of the background with no person in the picture. This picture is saved until the picture B (body) of the person is taken. Thus when processing of the picture of the body begins there are two pictures available to the program; both have been taken from identical camera angles and are the same size. One is a picture of the background, the other contains a man in front of the background.

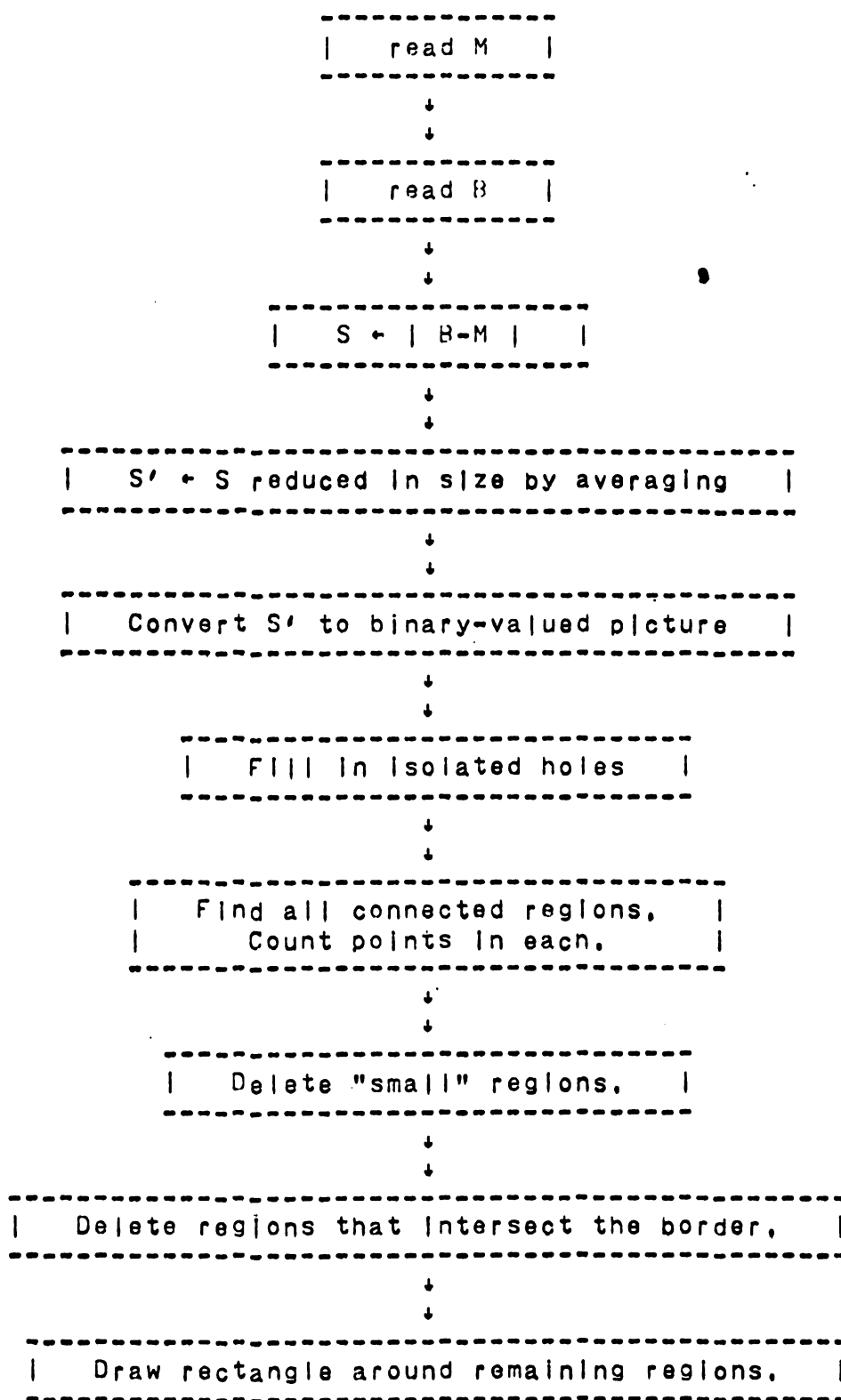


Figure 6-3.
FLOW CHART - APPROXIMATE LOCATION OF THE BODY.

In a very simple, yet very real sense, the picture M is a model of the background of picture B. If M and B are compared, those areas of B that are similar to M should be background. Those areas of B that do not match the model M are areas where a disturbance has entered the picture. These are the areas where the man is located.

The two pictures are subtracted. That is, a new picture S is formed with the intensity at each point of S equal to the absolute value of the difference of the intensities at corresponding points in M and B,

$$S_{ij} = |B_{ij} - M_{ij}|$$

In a simple world, S would be non-zero only at points on the man and at all of these points. In reality there are shadows and noise which produce false readings. In addition, some areas of the individual's clothes may be similar in intensity to the background and give no difference after subtraction. This may cause the non-zero areas of S to be subdivided into several parts or to contain holes. Therefore the first thing that must be done is to distinguish the blobs that make up the man from stray background blobs. This can be done using considerations of size and central location.

The subtracted picture S is reduced in size n times by averaging. The new picture is designated S'. The intensity at each point S'_{ij} is the average intensity over an $n \times n$ square of S. The averaging reduces noise; the size

reduction speeds up subsequent processing. Various values of n were tried. A large n speeds up processing but leads to considerable uncertainty in the location of the blobs that are identified. An n of 6 or 8 seems to be the most satisfactory compromise.

S'' is now converted to a binary valued picture S' by thresholding.

$$\begin{aligned} S'_{ij} &= 1 && \text{if } S''_{ij} \text{ is greater than } 1. \\ &= 0 && \text{otherwise,} \end{aligned}$$

The threshold value which is used, 1, insures that most points set to zero are truly background. An example of S' as it appears at this stage is given in Figure 6-2.

Isolated holes in S' are filled. An isolated hole is a point with value 0 that is surrounded on 4 sides with 1's. The value of such a point is set to 1.

In the picture S' , connected regions (subsets of the picture) are identified and labeled. Two points, S'_i and S'_n are connected if there exists a sequence $S'_1, S'_2, S'_3, \dots, S'_n$ such that S'_2 is a neighbor of S'_1 . Two points are neighbors if they are immediately adjacent horizontally or vertically; i.e., the neighbors of S'_{ij} are $S'_{i,j-1}$, $S'_{i,j+1}$, $S'_{i-1,j}$, and $S'_{i+1,j}$. A subset of the points of S' is a connected region if all points in that subset are connected. Connectivity, as defined above, is often called 4-connectivity (Rosenfeld [1970]).

The program that determines the connectivity of the

reduction speeds up subsequent processing. Various values of n were tried. A large n speeds up processing but leads to considerable uncertainty in the location of the blobs that are identified. An n of 6 or 8 seems to be the most satisfactory compromise.

S'' is now converted to a binary valued picture S' by thresholding.

$$\begin{aligned} S'_{ij} &= 1 && \text{if } S''_{ij} \text{ is greater than } 1. \\ &= 0 && \text{otherwise,} \end{aligned}$$

The threshold value which is used, 1, insures that most points set to zero are truly background. An example of S' as it appears at this stage is given in Figure 6-2.

Isolated holes in S' are filled. An isolated hole is a point with value 0 that is surrounded on 4 sides with 1's. The value of such a point is set to 1.

In the picture S' , connected regions (subsets of the picture) are identified and labeled. Two points, S'_i and S'_n are connected if there exists a sequence $S'_1, S'_2, S'_3, \dots, S'_n$ such that S'_i is a neighbor of S'_{i-1} . Two points are neighbors if they are immediately adjacent horizontally or vertically; i.e., the neighbors of S'_{ij} are $S'_{i,j-1}$, $S'_{i,j+1}$, $S'_{i+1,j}$, and $S'_{i-1,j}$. A subset of the points of S' is a connected region if all points in that subset are connected. Connectivity, as defined above, is often called 4-connectivity (Rosenfeld [1970]).

The program that determines the connectivity of the

picture is a procedure that is used several times in the processing described in subsequent chapters. It is an efficient procedure that determines the connectivity during one pass over the picture array. The labeling is given by a matrix of region numbers for each point and a table of equivalences. The algorithm is given in the form of an Algol procedure in Figure 6-4.

This algorithm was developed by the author. Subsequently, it was discovered that the underlying idea was first published in 1957 and has been re-invented several times since. (See the references in Rosenfeld [1969, p.138] as well as Lourie [1969]). However, none of these references give a detailed algorithm.

Now that connected regions in the picture are identified, it is easy to delete small regions and regions which intersect the border. A parameter, SM, determines what constitutes a small region. Four was found to be a satisfactory value for SM for the pictures in this thesis. This value of course depends on the resolution of the video input device.

At this point all that should remain in S are those connected regions which make up the body of the man. The extremes of these regions, horizontally and vertically, define a rectangle in which the man is located. All further processing is done only within this rectangle. The rectangle gives a first approximation to the size of the


```
procedure CONNECTED (A, REGNR, REGTBL, NREG, M, N, P);  
  value M, N, P;
```

```
  Boolean array A[0:M,0:N];
```

```
  Integer array REGNR[0:M,0:N];
```

```
  Integer array REGTBL[0:P];
```

```
  Integer NREG, M, N, P;
```

```
comment
```

This procedure labels connected regions in a binary valued picture. Four-connectedness is used. A is the picture matrix. To each "true" element of A a positive integer will be assigned such that all elements in a given connected region receive the same value, while elements in different connected regions receive different values. This positive integer is called the "region number". On exit from this procedure the region number for point A[I,J] is in REGNR[I,J]. The region number for any "false" point will be zero.

Also on exit, NREG will contain the number of distinct connected regions in the picture.

P, the length of REGTBL, must be at least as great as NREG.

For clarity, some of the administration required is omitted from this routine. There is no checking to see if REGTBL overflows and the "garbage collection" necessary to handle this overflow is omitted. Special case treatment of the picture borders is not included. Instead, it is assumed that for all I,

A[0,I] = A[I,0] = false

and REGNR[0,I] = REGNR[I,0] = 0;

Figure 6-4.
PROCEDURE CONNECTED.


```

begin
  Integer I, J, K, L, N1, N2, SMALL, LARGE;
  comment
    I, J, K, are running indices.
    L is the index of the last element of REGTBL
    in use.
    N1 and N2 are used to contain the current
    region numbers of the picture elements
    immediately above and to the left of A[I,J].
    SMALL and LARGE are temporaries;
  L := 0;
  REGTBL[0] := 0;
  for J := 1 step 1 until N do
  for I := 1 step 1 until M do
  if A[I,J] then
  begin
    N1 := REGTBL[REGNR[I,J-1]];
    N2 := REGTBL[REGNR[I-1,J]];
    if N1 = 0 ^ N2 = 0 then
    begin
      comment Both neighbors are zero so
      assign a new region number;
      L := L+1;
      REGTBL[L] := L;
      REGNR[I,J] := L;
    end;
    else if N1 = 0 then REGNR[I,J] := N2
    else if N2 = 0 v N1 = N2 then REGNR[I,J] := N1
    else
      begin
        comment Both neighbors are non-zero.
        Set up an equivalence between their
        region numbers;
        SMALL := if N1 < N2 then N1 else N2;
        LARGE := if N1 < N2 then N2 else N1;
        for K := LARGE step 1 until L do
        if REGTBL[K] = LARGE then
          REGTBL[K] := SMALL;
        REGNR[I,J] := SMALL;
      end;
    end;
    comment Count the regions;
    NREG := 0;
    for K := 1 step 1 until L do
    if REGTBL[K] = K then NREG := NREG + 1;
  end;

```

Figure 6-4. (continued from previous page)

man. This size information indicates such things as the size of the head to be searched for.

MEASURING HEIGHT.

The height of the person in the original picture is now measured. This is done by locating the top of the head and the feet and measuring the distance between them.

The top of the head is found first. The approximate location of the head is known from the previous processing of the reduced picture. It is still necessary to examine closely an area of the original picture to pinpoint the location of the top of the head.

The top of the head is found by a template matching operation. The reason that template matching works at this point is because the search region has been narrowed down to a small area. If an attempt was made to pass a head template over the entire picture many false responses would be encountered. This problem is minimized when searching a limited area.

A curved template has been made by considering a number of pictures of heads. Figure 6-5 is an example of such an "ideal" top-of-the-head. The size of the template is varied according to the approximate height of the individual which was obtained during the approximate location step.

The template is represented in the computer as a matrix of 0's, +1's, and -1's. A typical template is shown in


```

. . . . . . . . . . . . . . .
. . . . . . . . . . . . . . .
. . . . . 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
. . . . 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
. . 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
. 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
6 6 6 6 6 6 6 6 6 6 6 6 6 6 6

```

Figure 6-5. Ideal top of head.
 (" ," stands for "0" in the figures on this page.)

```

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 . . . -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 . . . +1 +1 +1 . . . -1 -1 -1 -1
-1 -1 -1 . +1 +1 +1 . . . +1 +1 +1 . -1 -1 -1
-1 -1 . +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 . -1 -1
-1 . +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 . -1
. +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 .

```

Figure 6-6. Template for top of head.

```

. 1 . . 2 . 1 2 . . . 3 . 1 1 . .
1 . . . 3 . . 2 . . . 1 . . 2 . .
. . 1 2 . . 4 6 5 4 6 3 . . .
. 2 . . 6 5 5 6 6 4 5 6 5 3 3 .
. . 6 9 9 9 6 6 5 6 5 6 5 5 5 .
. 9 12 11 12 10 6 5 6 6 7 6 5 6 6 5
11 12 10 12 10 12 11 12 10 6 5 6 6 7 6 6 5

```

Figure 6-7. Actual top of head.

Figure 6-6, This template is cross-correlated, in the area where the head is located, with the picture S (mentioned earlier) which contains the absolute value of the difference in intensity between the picture of the man and the picture of the background. In S it is expected that the top of the head will be somewhat like Figure 6-5. Above the head the values should be zero or close to it. Within the head the values should be positive.

As in all picture processing, noise and imperfections make the simple become difficult. If the picture were noiseless one could look for the step from zero background to non-zero head. However, the background is noisy and the interior of the head has steps of intensity within it that may be greater than the steps from background to head. Figure 6-7 gives a stylized example of these problems. The background has an average intensity of about 1. Most of the head has an average intensity of about 5. Within the head there is a curved area with average intensity near 11. In a straight-forward cross-correlation the step from 5 to 11 would give a higher response than the step from 1 to 5. This would be undesirable.

What is desired is to find the step from "low" background values to higher head values. In practice, it is not effective to define "low" background with a threshold for there is no natural way to set this threshold. Instead, the cross-correlation is computed using logarithms

to emphasize low steps of intensity,

The template T is matched with a rectangular subset of S as follows,

$$V_{i,j} = \sum_{p,q} T_{p,q} \ln(S_{i+p,j+q} + 1)$$

This is easily implemented efficiently because S is limited to sixteen grey levels, 0 to 15. The logarithms can be computed once and stored in a table for easy access during cross-correlation.

$V_{i,j}$ is the value of the cross-correlation at a particular point. The maximum $V_{i,j}$ found gives the location of the top of the head and the coordinates of the point are recorded.

Finding the feet is quite similar to finding the top of the head. An examination of Figure 6-1 shows that in the "standard" position the subject stands with heels together and toes apart. In this position, the outer sides of the toes form a characteristic pattern on the floor. Thus the feet can be found by template matching in the area where they are known to be approximately located. The logarithmic cross-correlation is used as with the head. The coordinates of the feet are recorded.

The height of the person is now known in picture units (raster units). This value is stored as the first measurement which characterizes the person.

MEASURING WIDTHS.

The width of the head, the neck, the shoulders, and the hips are now measured. Figure 6-8 shows where each of these measurements is taken.

The determination of the best place to take these measurements (e.g., the widest part of the head, the narrowest part of the neck) can be difficult. This difficulty is avoided by measuring at standard positions.

Let the measurements for width of head, neck, shoulders, and hips be designated by m_1 , m_2 , m_3 , and m_4 . Assume that the height of the subject is h raster units. Assume further that the feet are located at $(0,0)$ and the head at $(0,h)$ as in Figure 6-9. Then on a "typical" person the "best" place for measuring m would be at $(-x_c, y_c)$ and (x_c, y_c) where $x_c = w_c h$ and $y_c = r_c h$. The values used for w_c and r_c are given in Table 6-1. These values were obtained by hand measurements on a number of pictures.

Unfortunately, often the position of the body is not vertical. The TV camera or the vidicon tube may be displaced in such a way that the body appears at an angle. Accordingly, the search positions $(-x_c, y_c)$ and (x_c, y_c) must be mapped into correct positions (x_L, y_L) and (x_R, y_R) . The straight-forward calculations needed for this mapping are given in Figure 6-10.

Now, at each end of the four widths to be measured an edge detection operator is used to find the vertical edge.

Figure 6-8.
Measurements
of width.

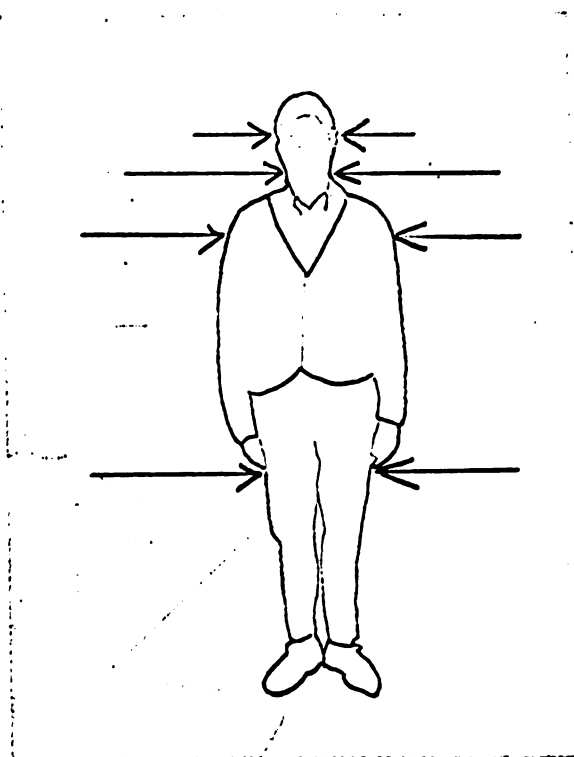


Figure 6-9.
Schematic diagram
of placement of
width measurements.

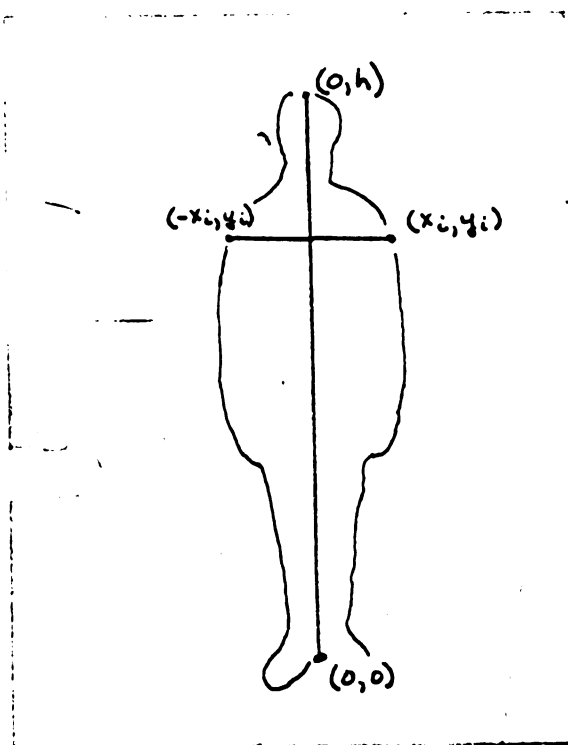
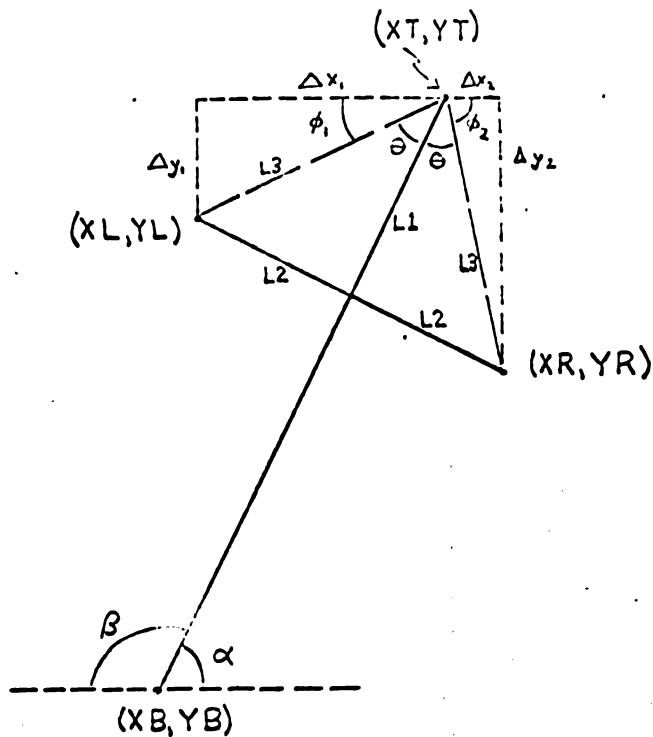


TABLE 6-1.
Standard positions for width measurements.

I	$w = x/h$	$r = y/h$	operator height
1	.11	.93	.028
2	.067	.87	.017
3	.28	.76	.028
4	.21	.38	.028



$$\beta = \arctan \left((Y_B - Y_T) / (X_B - X_T) \right)$$

$$\alpha = \pi - \beta$$

$$\theta = \arctan \left(L_2 / L_1 \right)$$

$$L_3 = \sqrt{L_2^2 + L_1^2}$$

$$\phi_1 = \alpha - \theta$$

$$\phi_2 = \beta - \theta$$

$$\Delta x_1 = -L_3 \cos \phi_1$$

$$\Delta x_2 = L_3 \cos \phi_2$$

$$\Delta y_1 = L_3 \sin \phi_1$$

$$\Delta y_2 = L_3 \sin \phi_2$$

$$X_L = X_T + \Delta x_1$$

$$X_R = X_T + \Delta x_2$$

$$Y_L = Y_T + \Delta y_1$$

$$Y_R = Y_T + \Delta y_2$$

Figure 6-10
Allowance for the variance from vertical.

The height of the operator is given in Table 6-1 as a fraction of the height of the subject. When these edges are found they provide the width in raster units of each feature.

The measurements are now recorded for use in the classification step. This concludes the processing of the picture of the entire body.

CHAPTER 7

FINDING THE OUTLINE OF THE HEAD

This chapter describes the program which has been developed for extracting an accurate outline of a man's head from a digital picture. A typical input picture is shown in Figure 7-1. When the methods described in this chapter are applied to the input represented by Figure 7-1, the result is the outline of the man's head as shown in Figure 7-2.

In the preceding chapter the methods used to extract measurements from the picture of the body were described. The resolution of that picture is not adequate for measuring the position of facial features. Therefore a second picture is taken, a close-up of the head of the subject. This picture, like that taken of the body, contains the head of the subject in front view, looking at the TV camera, against a background of normal room objects.

It is necessary to find an accurate outline of the head so that dependable reference points can be found from which the width of the head can be measured. The location of the top of the head is also obtained from the outline. The position of the outline of the head is used in subsequent processing to determine where to search for smaller features such as eyes.

The details of this processing are described in the remainder of this chapter. The picture presented in Figure

Figure 7-1.
Unprocessed input picture
of a man's head.

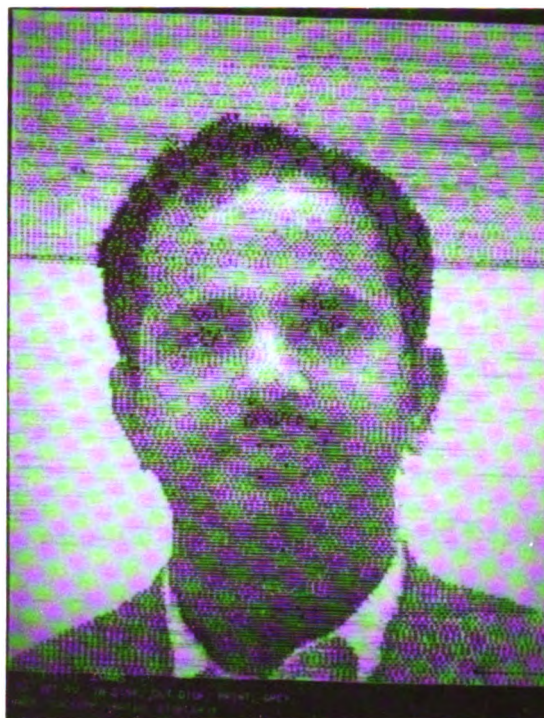


Figure 7-2.
The outline of a head,
the result of processing
Figure 7-1



7-3 will be used as an example throughout the chapter. The size of this picture is 226 x 325 points. In general the size of the pictures of the head is variable. Since the programs which search for and find heads is complex and only described in general terms in this chapter, a complete listing of these programs is included as Appendix 3.

REDUCE IN SIZE,

A new small picture is produced from the original. Each point in the small picture is the average value of the 64 points in an 8 x 8 square in the original picture. Figure 7-5 is an example of a small picture. Only the shape of the head is clearly visible. Other details have been smoothed out. The size of Figure 7-5 is 28 x 40 points. The decision to reduce in size by the factor eight was somewhat arbitrary. In smaller pictures the head tended to disappear; in larger pictures some unwanted details were still present.

FIND ALL EDGES,

The small picture obtained in the preceding step is processed to produce a new matrix which contains information on the edges in the small picture. This matrix will be called the small edge matrix. Figure 7-6 is an example of the small edge matrix derived from Figure 7-5. Each element in the matrix contains four bits, representing the four

Figure 7-3.
Unprocessed input
picture used to
obtain Figures 7-4
through 7-8

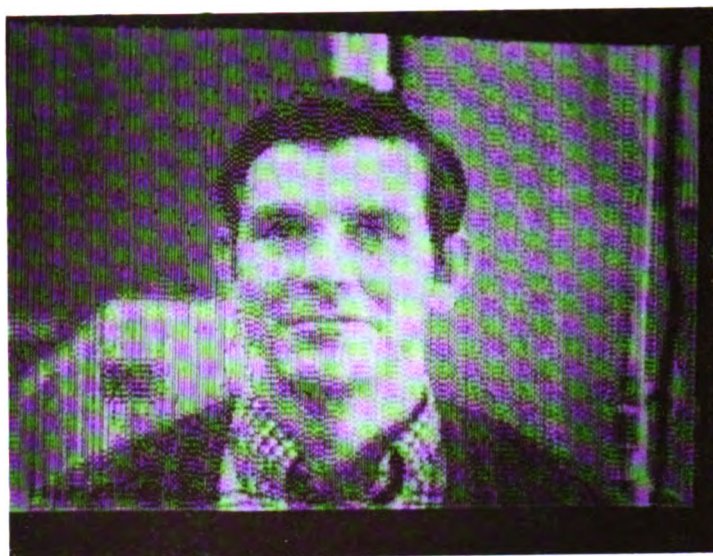


Figure 7-4.
The results of
applying a
gradient operator
to Figure 7-3



Figure 7-5.
The results of reducing
Figure 7-3 in size by averaging.



Figure 7-6

The results of applying
and edge-finding operator
to Figure 7-5.



Figure 7-7.

The "plan" extracted from
Figure 7-6 which will be used
to follow the head outline.



Figure 7-8.

The outline of the head
in Figure 7-3, obtained
using the plan of
Figure 7-7.



directions (horizontal, vertical, diagonal right, diagonal left) in which edges may be detected. All bits are zero if no edge was found at a point. A one in any bit indicates that an edge in that direction was found.

The local operator which is used to detect edges is described below. Many operators were candidates for this purpose, including those mentioned earlier in the section on edge detection. The choice of the particular operator used was made because of high confidence in its ability to reject false edges. Some edges may be missed but this is preferable to reporting false edges. A more straight-forward gradient operator was not used because they sometimes produce an edge indication in an uneven transition area of grey shades. In addition there is no natural cut-off threshold for the gradient operator. Hueckel's operator [1969], which can give very good results, was rejected because it was too large to be effectively used in the small picture. Most of the smoothing utilized by Hueckel's operator has already been done during averaging.

The edge operator used is applied to 3 x 3 squares of the picture. Let the points in this square be labeled as follows:

```

A B C
D E F
G H I,

```

The algorithm for detecting a horizontal edge is:

```

if B>H then
    if [min(A,B,C)-max(G,H,I)] > 1 then EDGE
    else NO-EDGE
else if B<H then
    if [max(A,B,C)-min(G,H,I)] > 1 then EDGE
    else NO-EDGE
else NO-EDGE.

```

The EDGE or NO-edge results apply to point E. This algorithm is also applied in the vertical direction (using points A,D,G and C,F,I) and along both diagonals (points A,B,D and F,H,I for one diagonal; points B,C,F and D,G,H for the other). Although this algorithm may appear to be time consuming, it may be programmed efficiently.

The small edge matrix is searched as described in the next step to find the small head outline. Because of

repeated search and backup during the search for the head, it is best to perform the local edge identification operation only once for each point. Since the picture is small, this does not consume excessive time.

FIND HEAD IN REDUCED PICTURE.

The search for the outline of the head is done in the small edge matrix. Various heuristics are included in the search program which define an acceptable head shape. The final result of the search is a list structure containing the coordinates of the points which constitute the outline of the head. Each entry in the list structure designates a point where an edge is present in a direction which is reasonable for that part of the head. An example of an unreasonable edge direction would be a vertical edge among a row of horizontal edges which have been labeled the top of the head. Figure 7-7 shows the edges of the head found in this step. The representation of acceptable head shapes is incorporated into program statements. There are searches, branches, and possible back-up as the outline of the head is built up. This process is similar in some ways to the S.R.I. decision tree search (Nilsson [1969]). If no head is present in the original picture, it is detected at this stage.

The details of the search for the head are as follows. Three short line segments are found which are candidates for

being part of the top and sides of the head. The spatial relationship between these lines must be reasonable (e.g., the top of the head must be above the sides). An attempt is made to connect these line segments to form a "head" shape. The program searches the region between them for edges which are part of the somewhat semicircular top of the head. If this cannot be done, other possible short line segments are tried as sides or tops. If the top half of the head is found, a search is made below it for the inward curves of the neck and then the curves outward toward the shoulders. When all of these requirements have been met, the head has been found. The sides of the head are then examined for indentations where the ears should be since the ears sometimes merge with the background. If indentations exist, these are filled in.

USE PLAN TO FIND OUTLINE IN ORIGINAL.

This part of the program is a plan follower. Its input is the full size intensity picture and the list structure containing the small head outline. The output of the plan follower will be a new list structure containing the coordinates of an accurate outline of the head. Between successive points in the plan, the plan follower searches a narrow band for an edge which connects the points. This band is sixteen points wide, since the plan was reduced in size by a factor of eight. Although this band is narrow

compared to the size of the picture, it is still wide enough to contain several edges. The proper edge is chosen primarily by direction. If there are two parallel edges running in the desired direction within the search band, that edge farthest from the center of the head is chosen. The reason for this choice is the assumption that edges within the head (in hair, ears, etc.) are more likely to follow the directions of the head outline than edges found on background objects.

The operator used to detect edges in the full sized picture is almost the same as that used in the small picture. In the small picture the operator was applied to the nine points of a 3 x 3 square. In the large picture the same operator is applied to the nine points at the corners, the center of the sides, and the center of a 5 x 5 square, e.g.

```
A - B - C
- - - - -
D - E - F
- - - - -
G - H - I
```

The reason for this change was to allow detection of faint edges in the large picture.

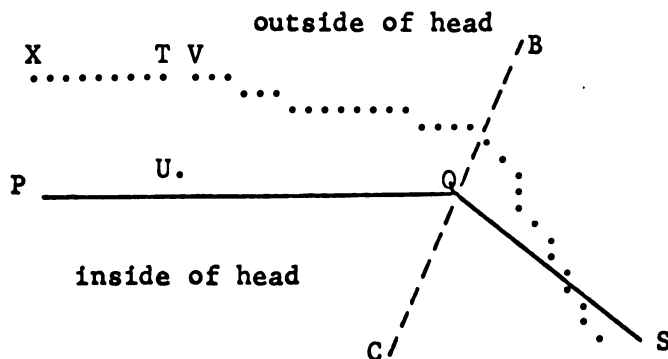


Figure 7-9. Operation of plan follower.

The details of the plan follower are given below. Reference is made to Figure 7-9 which is a schematic diagram of the operation of the plan follower in a small region. Two non-parallel straight lines are identified in the plan, lines PQ and QS. Point X is the last previous point that has been found on the edge. The direction of search is generally from P to Q to S. The line BC is found which bisects the angle PQS. Corresponding to the line PQ in the plan, the plan follower will search from point X until the edge crosses line BC. Within this region the edge will be accepted if its direction is roughly parallel to PQ or to QS. The edge search is done by moving one unit in the direction of PQ from the last point found and applying the

edge detection operator along a line perpendicular to PQ. Normally the edge detection operator is only applied to five points immediately in front of the last point found. Under certain conditions however the edge detection operator is applied over all 16 points across the search band. These conditions are:

- a. The edge has just taken a sharp turn.
- b. The edge is lost.
- c. The intensity outside the head outline has changed abruptly.
- d. The edge zig-zags.

These measures are necessary to correctly track the edge when it is sharply curving. The three points most recently found are filtered to remove a single point which is far off the edge. For example, if U were the only edge point found between points T and V, point U would be rejected.

SPEED COMPARISON.

The search for the outline of the head could have been done without using planning, as pointed out in Chapter 5. Tests were made to determine the speed improvement obtained by using planning. The results of these tests are given in Table 7-1.

These results indicate that using planning is about 40 times faster than the same search without planning. The tests were performed as follows. First the program just

TABLE 7-1.
Planning speed comparison.

PLANNING		TIME
STEP		
1.	Reduce in size.	2.5 sec.
2.	Find all edges.	.8 sec.
3.	Find plan (small outline).	.3 sec.
4.	Use plan (large outline).	2.6 sec.
Total		6.2 sec.

WITHOUT PLANNING		TIME
STEP		
1.	Smooth picture.	160. sec.
2.	Find all edges.	34. sec.
3.	Find head outline.	40. sec.
Total		234. sec.

described which finds the head outline using planning was timed. The time to process a head picture from input to completion of the large outline was about 6.1 seconds. This time, as well as all others mentioned, was measured on the PDP-10 computer and is accurate to $\pm 20\%$. Then the program was modified to perform without planning.

The first step was to eliminate the fine detail of the picture by smoothing in a fashion similar to the averaging used in planning. The entire picture was smoothed using averages over 8×8 squares. This took 420 seconds. By optimizing the program, this time could be brought down to 64 times the speed without planning, namely 160 seconds. This latter time is used in Table 7-1.

All edges were found in the smoothed picture. This took 34 seconds.

Next the search program was used on the full sized smoothed picture to search for the outline of the head. Here is where planning really showed its speed. This search, in a reduced size picture, takes about 0.3 seconds. In the large picture, the search took about 40 seconds. The reason for this is that quite a sizeable incorrect data structure is built up when following false paths and many points must be processed before the error is detected.

In summary, the program without planning took 494 seconds, 80 times slower than the time using planning. By optimizing the program, this could be improved to about 234

seconds. This would still be 40 times slower than planning. This comparison cannot be complete without again stating the primary advantage that planning gives to the designer of a program which searches for edges or objects: a small picture that he can comprehend both locally and globally.

LOCATE MEASUREMENT POINTS.

Now that the outline of the head has been found it is necessary to locate the top of the head and the points on the sides from which the width of the head is measured. This is done using a sliding average to avoid single point errors which may be present in the outline.

The outline of the head is represented in the computer by a list structure. Each element of the list contains the two coordinates of a point in the outline as well as pointers to the list elements representing the adjacent points on both sides. Thus it is possible to traverse this list in either direction. There are also a number of pointers external to the list which mark approximate locations of key elements.

These key element pointers have been obtained in the following way. Originally a plan for the head was formed, as already described. This plan was created using the known structure of the head. So in creating the plan certain areas were identified as top of head, extremes of approximately vertical side of head, etc. This information

is carried forward to the plan following stage. When the plan follower passes one of these areas while creating the list structure for the true head outline it sets an appropriate pointer into the list structure.

Knowing the approximate location for the top of the head, it is straight-forward to search for the highest point in the outline in this area. A sliding average of five points is used. The highest value gives the coordinates of the top of the head.

In a similar way the widest part of the head is identified on both sides of the head. The difference between these values is recorded as the width of the head. The coordinates of the points which have been identified on the outline of the head are now used to guide the search for the features interior to the head.

CHAPTER 8

FINDING THE FEATURES OF THE FACE,

The previous chapter described how the outline of the head was located. Once this outline is found, approximate locations for the eyes, the nose, and the mouth can be predicted. In spite of the fact that the approximate location of these features is known, there is still considerable processing that must be done to locate them precisely so that measurements of position may be made. This chapter discusses the methods used for finding each feature.

It is appropriate here to describe the coordinate system used in processing pictures in this program, since much of the discussion which follows makes use of this coordinate system. A left-handed coordinate system is used with the origin at the upper left of the pictures. (See Figure 8-1.) There are two reasons for this. First, this is the method used by the system when setting the limits on a window to be read by the TV camera. Secondly, it provides a convenient notation for interchanging coordinates (x,y) of a point with matrix indices (i,j) for that point.

EYES.

The measurement that is desired from each eye is the location of the center of the iris, the dark circle in the

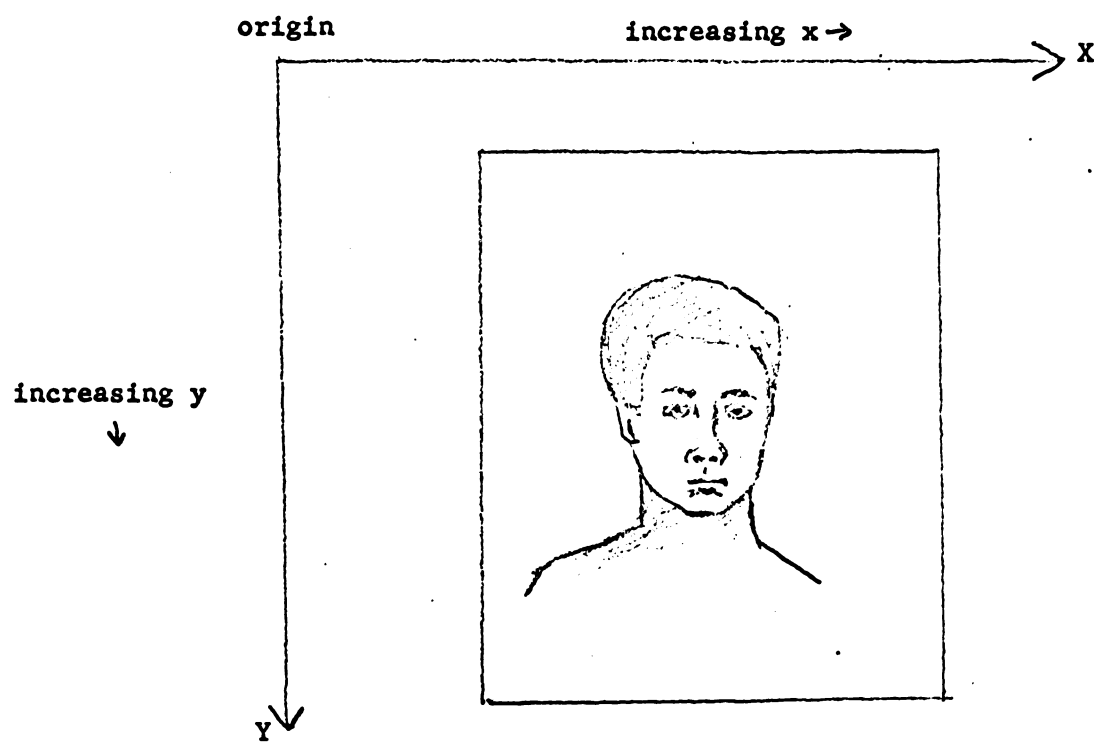


Figure 8-1.

Left-hand coordinate system used.

center of the eye. The iris is located by finding horizontal cross-sections which exhibit the characteristic shape shown in Figure 8-2. In essence, this is a template matching operation. The shape of Figure 8-2, when plotted as light intensity vs. x coordinate, possesses the following features. Outside the eye the skin has an irregular intensity of medium value. The white of the eye on both sides of the iris forms high peaks of light intensity. The iris itself is a dark valley between these two peaks.

The characteristic cross-section of the eye is elusive and difficult to find. The peaks and valley that are sought can be nothing more than anthills and a depression. To find them, one must know where to look. Pinning down just where to look for the iris and the whites of the eyes constitutes the bulk of the processing used on the eyes. The method used is called dynamic threshold setting.

Approximately locating the eye is also difficult. The detail in this area, as well as shadows around the eyes and nearby hair, can confuse the program. The one thing that can be found is a characteristic dark blob formed by the dark iris, eyebrow, and shadows under the overhanging brow. However, there is no natural threshold which defines the dark region. What is meant by dynamic threshold setting is the process of searching for a threshold that clearly shows this dark region.

The first thing that is done is to predict the location

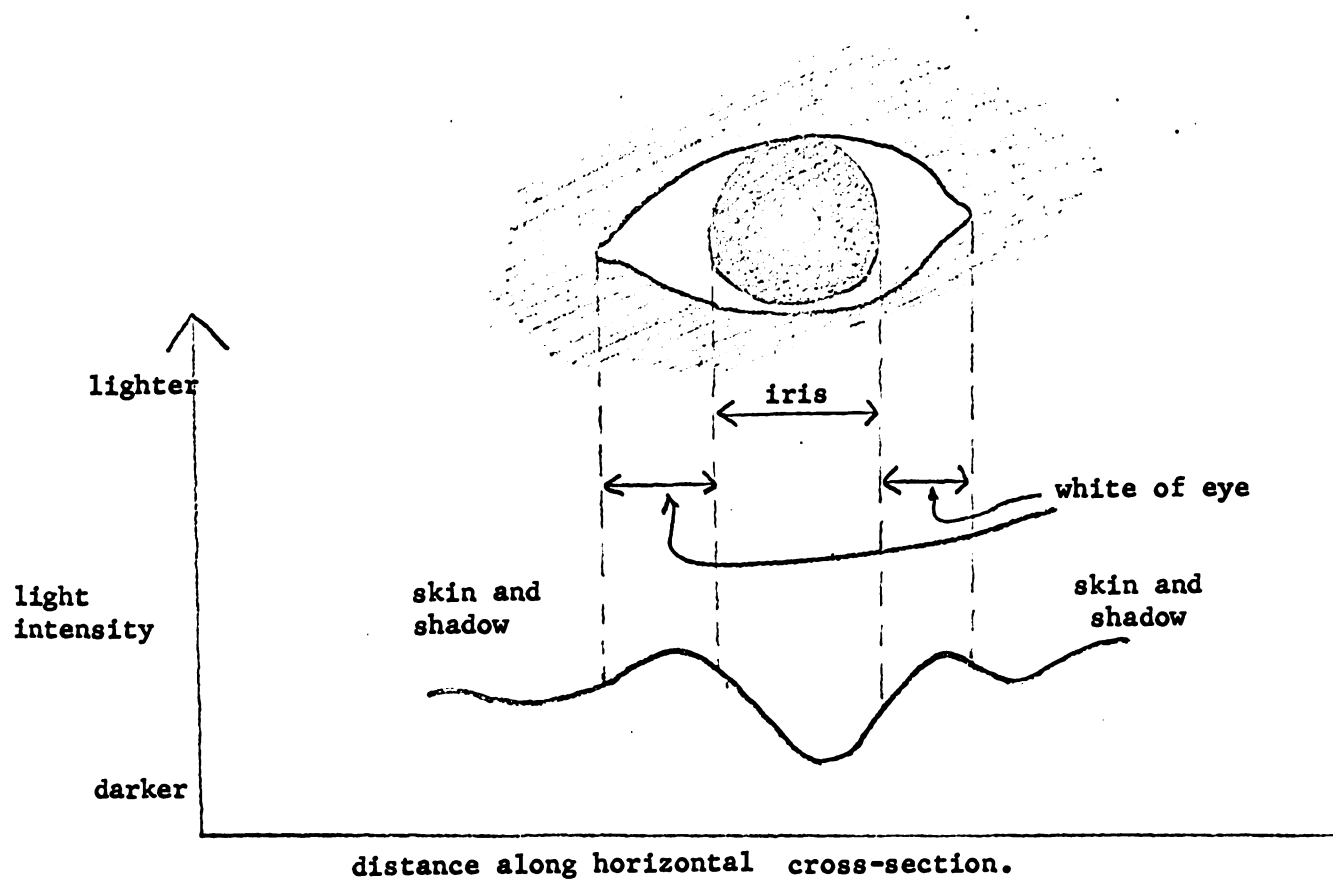


Figure 8-2. Horizontal cross-section which characterizes the eye.

of the eyes. This prediction is based on the average position of the eyes within the head. The predicted eye location will be used as the center of the area to be searched for the eye.

Figure 8-3 shows the model of the head and eyes on which the prediction is based. According to this model, if the top of the head is on the line $y=0$ and the sides are on the lines $x=0$ and $x=W$ (for width), then the eyes are located at

$$(FX(1)*W, FY(1)*HWRAT*W) \quad \text{and}$$

$$(FX(2)*W, FY(1)*HWRAT*W).$$

The experimentally obtained values of the parameters of this model are given in Figure 8-4.

Figure 8-5 shows how this model is used to predict eye location in an actual picture. From the previous processing it is known that the left and right sides of the head are at $X=XL$ and $X=XR$. The width of the head is therefore $W = XR - XL + 1$. Using the model, the x coordinates of the two eyes should be

$$X = FX(1)*W + XL \quad \text{for left eye}$$

and
$$X = FX(2)*W + XL \quad \text{for right eye,}$$

The top of the head is known to be at $Y=YT$. So from the model, the y coordinate of the eyes can be predicted as

$$Y = FY(1)*HWRAT*W + YT.$$

The predicted eye location is only a rough first approximation to the actual location of the eyes. This

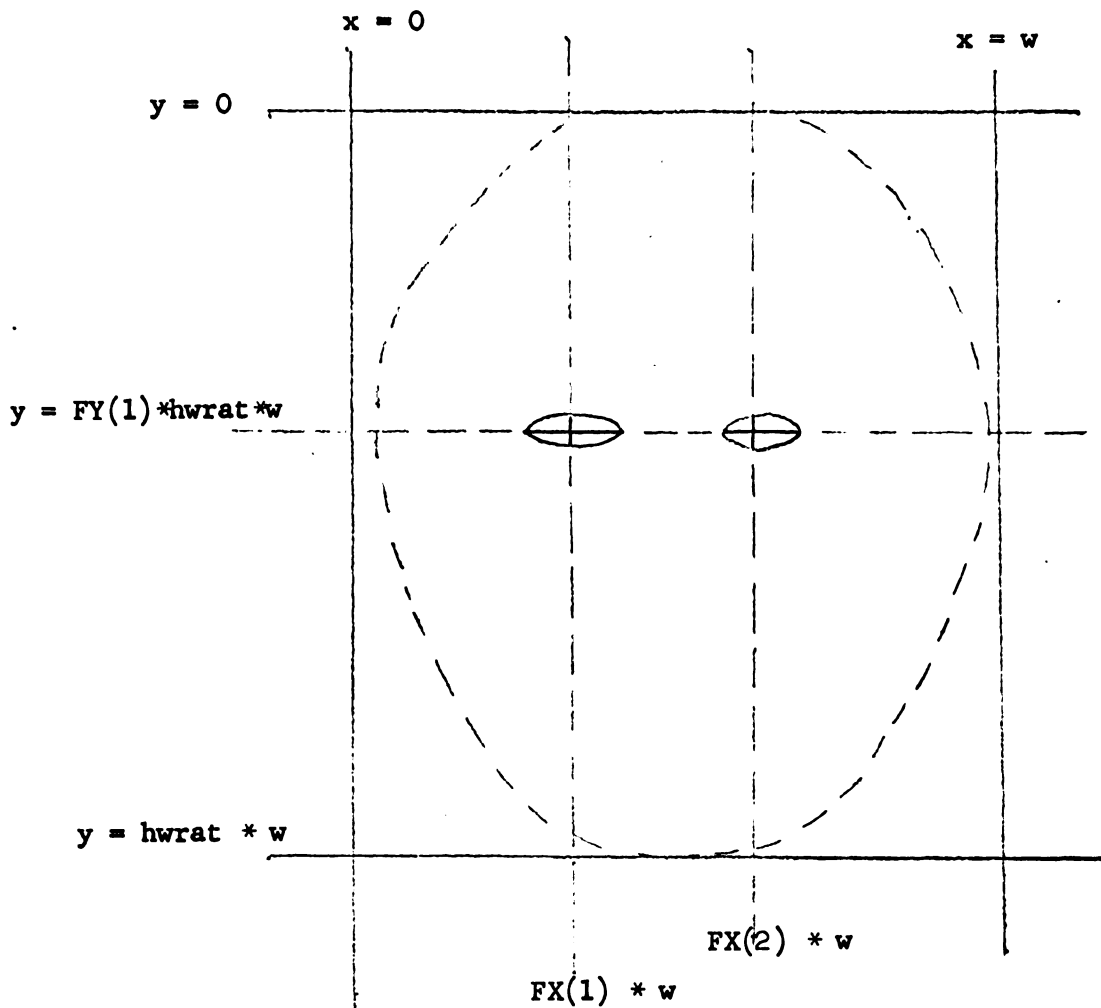


Figure 8-3. The model for eye location on an average head.

FX(1)	.329
FX(2)	.671
FY(1)	.494
HWRAT	1.2

Figure 8-4. Constants used in predicting the location of the eyes.

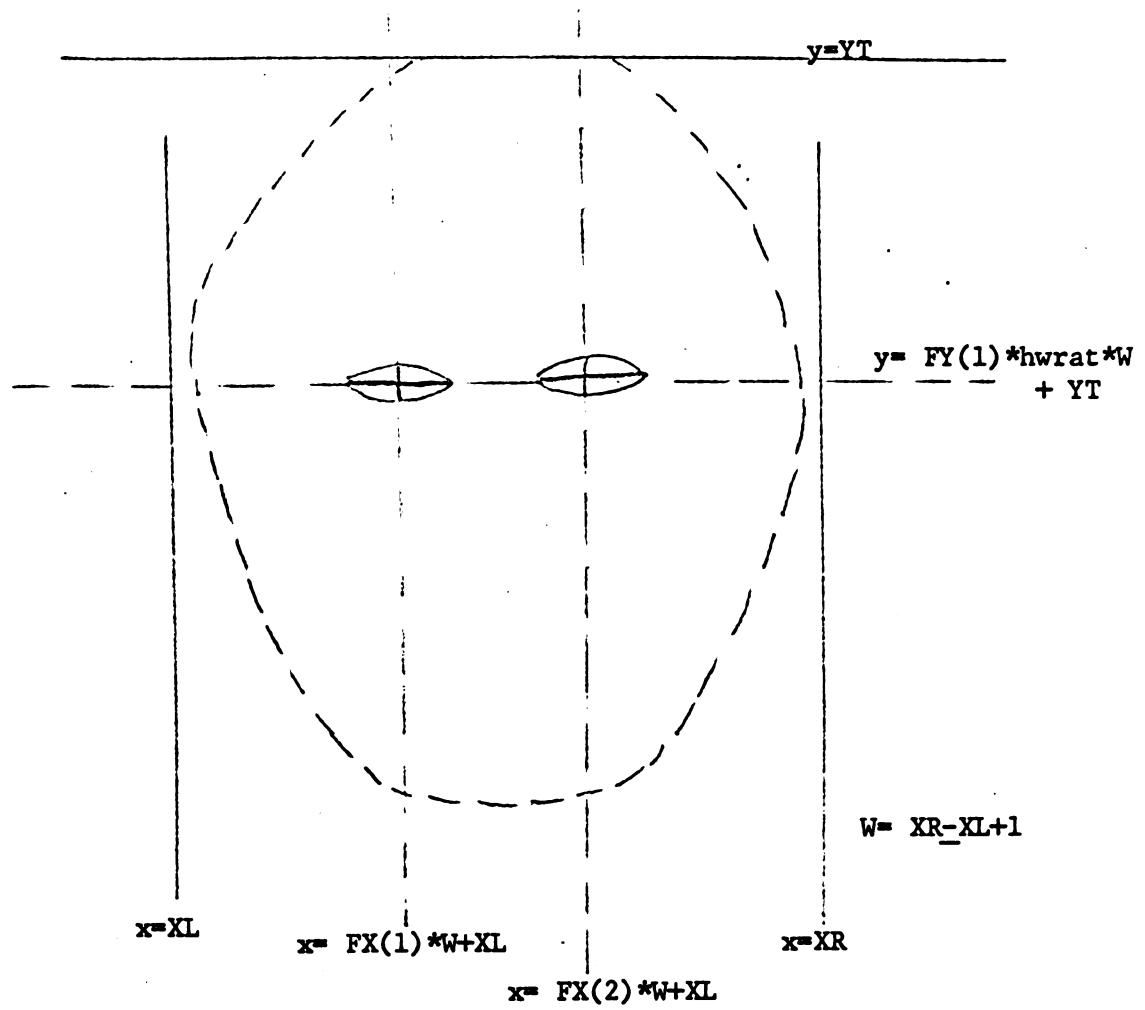


Figure 8-5. Predicting the location of the eyes on a given head.

approximation must be refined considerably before the search for the iris can be expected to work.

The area around the eye is found by looking for a dark blob. The iris, the lashes, the eyebrows, and especially the shadows under the overhanging eyebrow form an area that is darker than the surrounding skin. When this dark area is located, it will be possible to look within it for the iris. Locating this dark area is the next step in eye processing.

A search procedure that examines first a large area, then a small area, is used to home in on the dark blob that is the eye. The reasons for this two part search are given in the discussion of step A4 which follows. The flow chart of this search procedure is given in Figure 8-6. The steps of this procedure are discussed in detail in the following paragraphs.

A1. A square region to be searched for the eye is selected. This square is centered on the predicted eye location. It is large enough so that it will quite surely contain the eye even though the eye may be fairly far away from the predicted eye location. A suitable size for this square was determined experimentally. The size used is based on the measured width of the head W and is $.383 W$.

A2. Within this square, points which are dark and within large, centrally located, connected regions are identified by dynamic threshold setting. The procedure that identifies these points is fairly complex. It is used

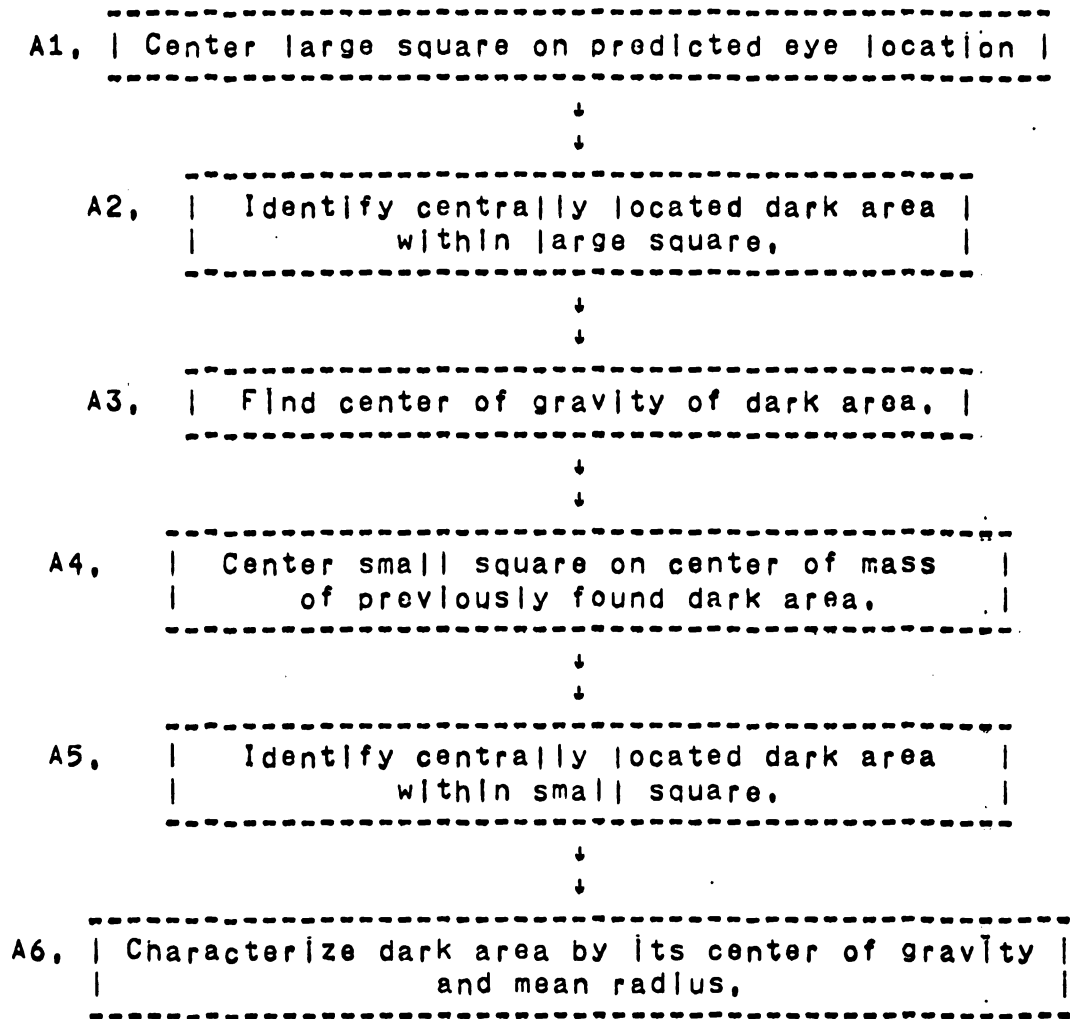


Figure 8-6. Procedure for finding the dark area around the eye.

at steps A2 and A5, It will be described in detail following the completion of the discussion of the flow chart given in Figure 8-6. In general, however, the procedure looks for a dark area that surrounds the eye. What is meant by "dark"? It is not an absolute measure. Rather, when examining the square under consideration, some points are darker; some are lighter. Some fraction of the points must be heuristically defined as "dark". This fraction is a parameter that was determined experimentally. The value used is .107. The procedure omits dark points which are isolated; it assumes these represent noise. It also omits all dark points which are part of connected regions which intersect the border of the square. These are assumed to be hair, the other eye, or shadows on the nose. Because the search square is large, these other features are sometimes present.

A3. The dark points found in the preceding step should be the points that make up the dark area around the eye. The center of this area is defined to be the centroid of these points. Let S be the set of "dark" points found. Let n_s be the number of points in S . For $\alpha \in S$ let the coordinates of α be x_α, y_α . Then the centroid $(\bar{x}_1, \bar{y}_1)$ is computed from

$$\bar{x}_1 = \frac{\sum_{\alpha \in S} x_\alpha}{n_s}$$

$$\bar{y}_1 = \frac{\sum_{\alpha \in S} y_\alpha}{n_s}$$

A4. It might be expected that the dark area found above is an acceptable approximation to the dark area around the eye. This is not the case. The extent of the area found is strongly dependent on the fraction used to define "dark" points. Since the square used must be quite large so that the eye will not be missed, the uncertainty in the value of this fraction is large. Another iteration of the dynamic threshold setting procedure is used. This time a smaller square will be searched so that the confidence in the value of the fraction is higher. The square will be centered on the centroid of the dark area which was found previously. The size of the square used is again based on a proportion of the width W of the head. The side of the square is $.287 W$.

A5. The centrally located dark area within the small square is identified. The procedure is the same as that used in step A2. The value of the fraction used to define dark points is $.06$.

A6. The dark points found in step A5 are characterized by their centroid as was done in step A3. The new centroid is denoted by $\bar{x}_2, \bar{y}_2$. An additional parameter is measured which characterizes the dark region. This is the "mean radius", $\bar{r}$, of the dark region. Let S , n_s , α , x_α , and y_α be defined as in A3. Then the mean radius is given by

$$\bar{r} = \frac{\sum_{\alpha \in S} [(x_\alpha - \bar{x}_2)^2 + (y_\alpha - \bar{y}_2)^2]^{1/2}}{n_s}$$

This completes the discussion of the general flow given in Figure 8-6. The dark area around the eye has been found and characterized by its centroid and its average radius. A picture of this characterization is given in Figure 8-8.

Before going on to discuss locating the iris, the details of the dynamic threshold setting procedure used in steps A2 and A5 will be given. A generalized flow chart of this procedure is given in Figure 8-7.

The procedure attempts to find a dark area that is central to a square subset of the picture. The dimensions of the square are input parameters to the procedure. Another input parameter is NDARK, the number of dark points expected. The values of these parameters were specified in the section which described the calls on these procedures.

An inspection of Figure 8-7 shows that there are several loops within this procedure. The looping may not be necessary in pictures where the eye region is clearly distinct from its surroundings. First, the flow of the procedure will be discussed assuming a straight, non-looping flow through the program. Following this, the reasons for allowing iteration and the loops will be discussed.

Let A_{ij} represent the grey-valued picture matrix which includes only the points in the square under consideration. The first step is to form a histogram of the intensities of S . The histogram is formed in the array $h[0:15]$. The value of the elements of h are

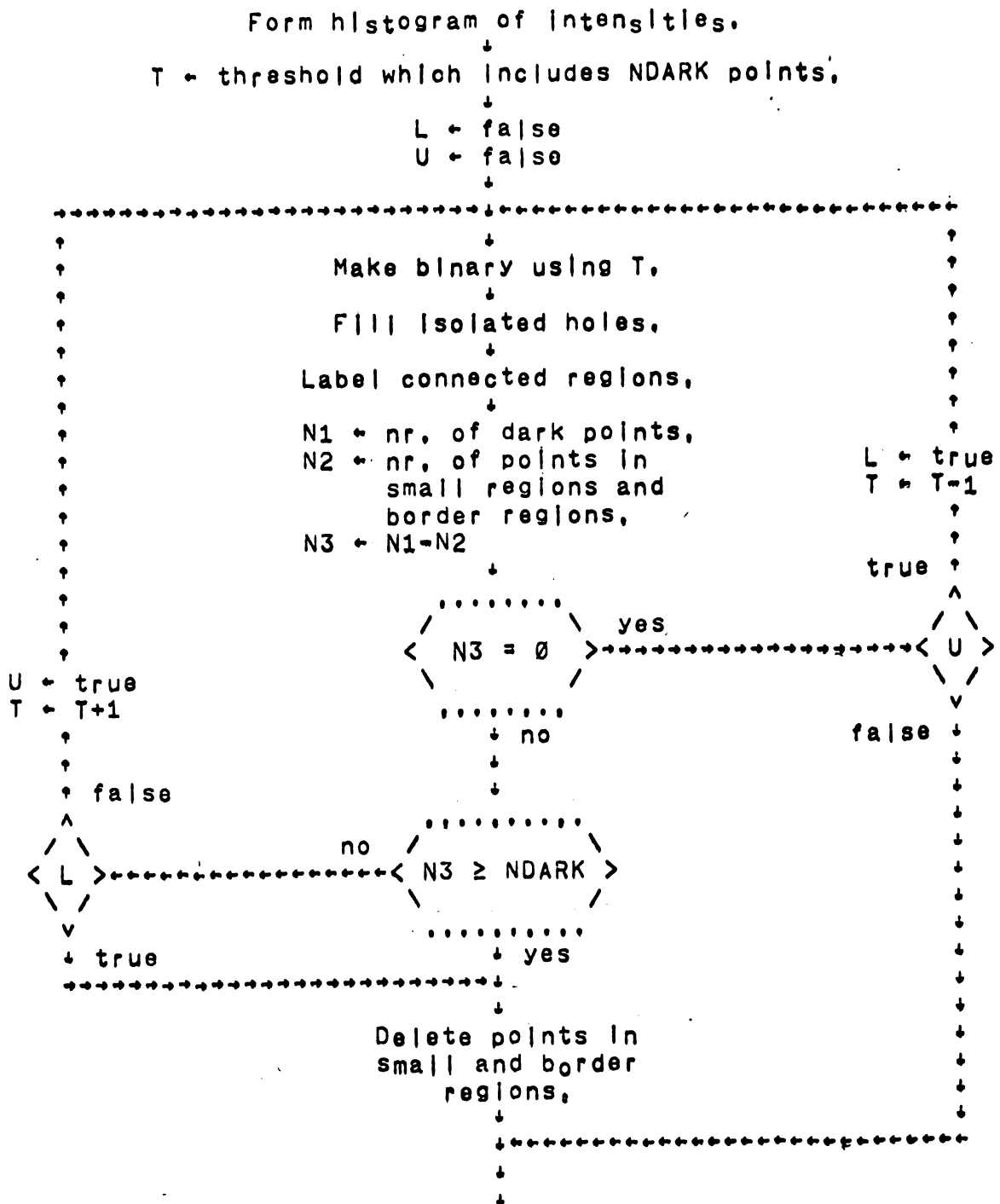


Figure 8-7. Dynamic threshold setting.
(Procedure for finding central dark areas.)

Figure 8-8. The dark region around the eye and characterizing parameters.

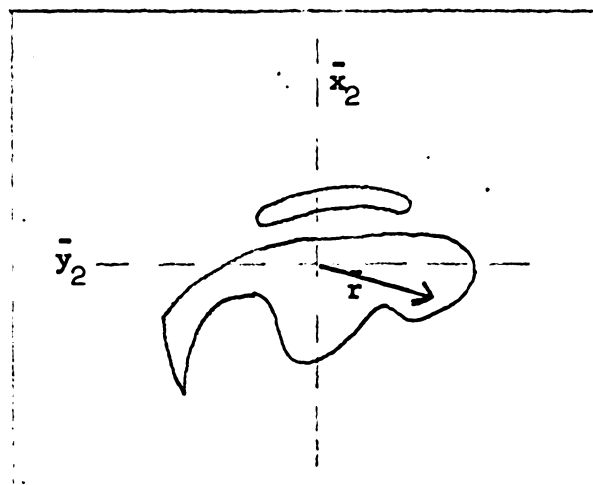
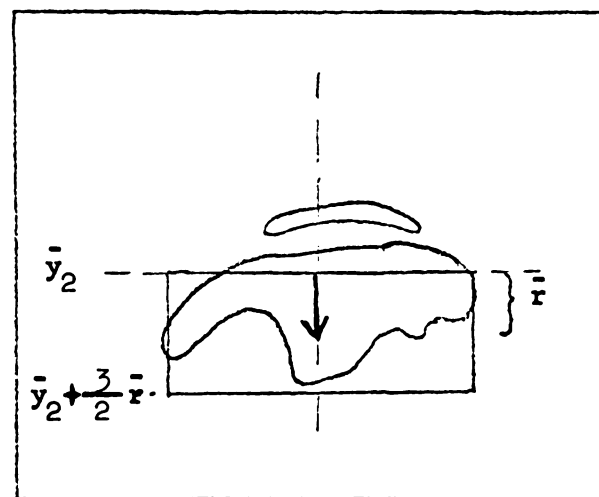


Figure 8-9. The area searched for the iris.



$$h_k = \sum_{i,j} [\text{if } A_{ij} = k \text{ then } 1 \text{ else } 0],$$

The variable T is set to that threshold that includes NDARK points. More precisely, T is the minimum value such that

$$\sum_{k=0}^T h_k \geq \text{NDARK}.$$

It is clear that T divides the picture A into two sets of points: the dark points with $A_{ij} \leq T$ and the lighter points with $A_{ij} > T$.

Now two Boolean variables, L and U , are set to "false". These are only used if looping is necessary. This completes the initialization of the procedure.

The main body of the procedure begins with construction of a binary-valued picture B from A . The threshold T is used in this construction as follows:

$$\begin{aligned} B_{ij} &= 1 \text{ if } A_{ij} \leq T \\ &= 0 \text{ if } A_{ij} > T. \end{aligned}$$

Thus the 1's in B represent the dark points of A .

The picture B is smoothed somewhat by filling in isolated holes. Points with value 0 are set to 1 if they are surrounded with 1's.

Connected regions of 1's in B are identified and labeled. This is done using the procedure "CONNECTED" which was discussed in detail in Chapter 6 (see Figure 6-4). Since each dark point of B is labeled with the region number of the connected region to which it belongs, it is easy to count the number of points in each connected dark region and

to identify those dark regions which intersect the border.

N_1 is set equal to the total number of dark points in B . N_2 is set equal to the number of points in B that are in small regions (< 4 points) or in regions that intersect the border. N_3 is set to $N_1 - N_2$. N_3 , then, is the number of points in connected dark regions that are reasonably large and somewhat central.

N_3 is tested to ensure that it is greater than $NDARK$. If it is, all is fine. Those points previously identified as being members of small regions or border regions are deleted from the binary picture B . When a point is deleted, its value is set to zero.

The procedure has completed its work. The binary picture B is available with 1's only in points that are central and dark. The centroid and mean radius can be computed using B .

Now, we must return to consider the cases where iteration is necessary in the procedure. This is necessary if fewer than $NDARK$ points would be left after deleting small regions and border regions. Recall that N_3 is the variable that is a measure of these points. There are four cases to consider.

Case 1. $0 < N_3 < NDARK$ and L is false. In this case the assumption is made that the threshold T was not set high enough. It was desired that $NDARK$ points should be present in the eye area. When much dark hair from the forehead or

temples enters the square under consideration, the threshold T , which was obtained from the histogram during the initialization of the procedure, is unreliable. The dark points from the hair are present in the histogram but then are deleted as border points. The solution is to raise the threshold T . Although more hair may be included it will be ignored since it always extends to the boundary of the area under consideration. T is increased by 1. The Boolean variable U is set to true to indicate that the threshold has been moved up. A transfer is made to the beginning of the body of the procedure.

Case 2, $N3 = \emptyset$ and U is false. If U is false, this is the first iteration of this procedure. At the first threshold setting, all points appear to be border points or in small regions. When this condition is reached, it indicates that the eye region possesses a strange connectivity in the picture under consideration. All dark points are kept in the matrix B and the program is allowed to proceed onward to the next stage of processing. Hopefully, the tolerance of the remaining routines will be able to handle this strange picture. (This case is included in the program only for completeness. It has never been encountered when processing normal pictures.)

Case 3, $N3 = \emptyset$ and U is true. In this case we have already encountered case 1 and have increased the threshold. In doing so the entire set of dark points has become

connected to the borders. This undesirable situation is remedied by lowering the threshold one notch and using those results as the best available. T is decreased by one. L is set to true. The program transfers to the beginning of the body of the procedure. Case 4 will occur on the next iteration.

Case 4, $0 < N3 < NDARK$ and L is true. The previous iteration encountered Case 3. T was raised too high. Accept the value of $N3$. Delete the small regions and border regions. The resulting matrix B is suitable for further processing.

This completes the description of the dynamic threshold setting procedure. We will now consider the routines which detect the iris.

Figure 8-8 shows the dark region around the eye and the parameters, $\bar{x}_2, \bar{y}_2$, and $\bar{r}$, which characterize the region. Now it is necessary to find the iris. In particular, we wish to find the characteristic horizontal cross-section that was shown in Figure 8-2. An effective procedure for doing this is the following.

First the area where the iris is expected to be found is heavily smoothed. This area is bounded above and below by the lines $y = \bar{y}_2$ and $y = \bar{y}_2 + \frac{3}{2}\bar{r}$ and on the ends by the maximum extent of the dark region. This area is illustrated in Figure 8-9. Within this area smoothing is done by replacing the intensity at each point with the average

Intensity over a $p \times q$ rectangle centered at the point. The height of the rectangle, p , is equal to the mean radius $\bar{r}$. The width, q , is $\frac{1}{3}\bar{r}$. A typical value for $\bar{r}$ is 7. Thus a typical size for the smoothing rectangle is 7×2 .

Now, a horizontal cross-section is taken along each line in the rectangle in Figure 8-9. The characteristic shape (two mountains of eye white with a valley between for the iris) can be identified. The center of the iris is located, and its coordinates are recorded.

The positions of the eyes have been located. Therefore, processing of the eyes is complete.

NOSE.

The measurements that are obtained from the nose are the coordinates of the center of the two nostrils. In a front view of the face the nostrils are the only part of the nose that is reliably and consistently present.

The nostrils appear in the digital pictures as two small dark areas of roughly elliptical shape. A typical nose region may be seen in Figure 8-10.

The processing needed to locate the nostrils is similar in concept to that used in processing the eyes. First, a location for the nostrils is predicted, based on a model of the relationship of the eyes and nose. Then the area surrounding the predicted location is searched to determine the actual location.

Figure 8-10.
Nose region in
unprocessed picture.

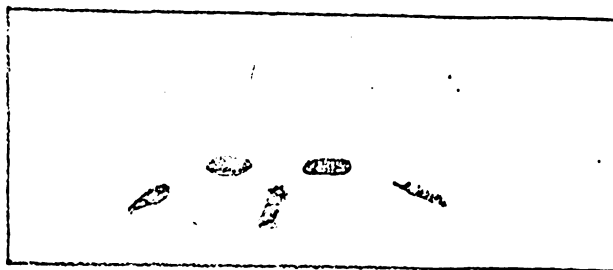


Figure 8-11.
Essentials of the
model of the eye-nose
relationship.

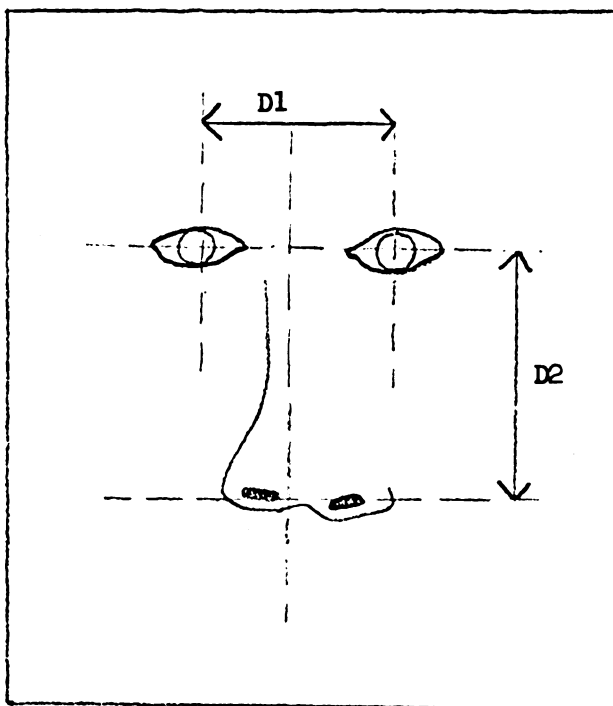


Figure 8-11 shows the essentials of the model that relates the positions of the eyes and the nose. The distance between the eyes is denoted by $D1$. The distance between parallel horizontal lines intersecting the eyes and intersecting the nostrils is denoted by $D2$. In the model $D2/D1$ is assumed to be a constant ratio. The experimental value used is .75. The model is used as follows. Let XL, YL and XR, YR be the previously measured eye locations. Then the predicted nostril locations XP, YP are given by

$$XP = \frac{1}{2}(XL + XR)$$

$$YP = \frac{1}{2}(YL + YR) + \left(\frac{D2}{D1}\right)(XR - XL).$$

These coordinates give a first approximation to the location of the nostrils. A square centered on XP, YP will be searched for the nose. The sides of the square are .287 times the measured width of the head W .

A key point in locating the nose, as in finding almost every other feature, is preliminary smoothing. This tends to make prominent things prominent and to make the minor local variations have a smaller effect. The smoothing is done on the nose region by replacing the average intensity at each point in the square under consideration by the average intensity of the four points in a 2×2 square.

At the same time that this smoothing takes place, a histogram of intensity is made. This histogram is used for subsequent threshold setting.

The nostrils are located by a procedure which does

dynamic threshold setting. The broad outlines of the search for the nostrils is given by the flow chart in Figure 8-12. This flow chart is discussed in detail below. The general idea is as follows. A threshold T is gradually raised from lower to higher grey values. At the lowest value only the few darkest points are below the threshold. As the threshold is raised a little, several small dark areas will be below the threshold. These areas should include the two nostrils. There may also be dark areas for shadows under the nose or from the sides of the nose. A heuristic decision procedure is used to identify those dark areas which represent the nostrils.

The details of the procedure of Figure 8-12 are as follows.

C1. The first value of the threshold T is chosen as the minimum intensity in the nose region. This value is obtained from the previously prepared histogram.

C2. Dark regions are identified. A dark point is a point with intensity less than or equal to T . A binary valued picture is formed with dark points having value 1, and light points having value 0. This is similar to the procedure used in locating the dark eye area. The procedure for identifying connected regions (CONNECTED, Figure 6-4) is applied to the binary valued picture; its methods were discussed previously in eye and body processing. Each dark point is associated with the region number of the connected

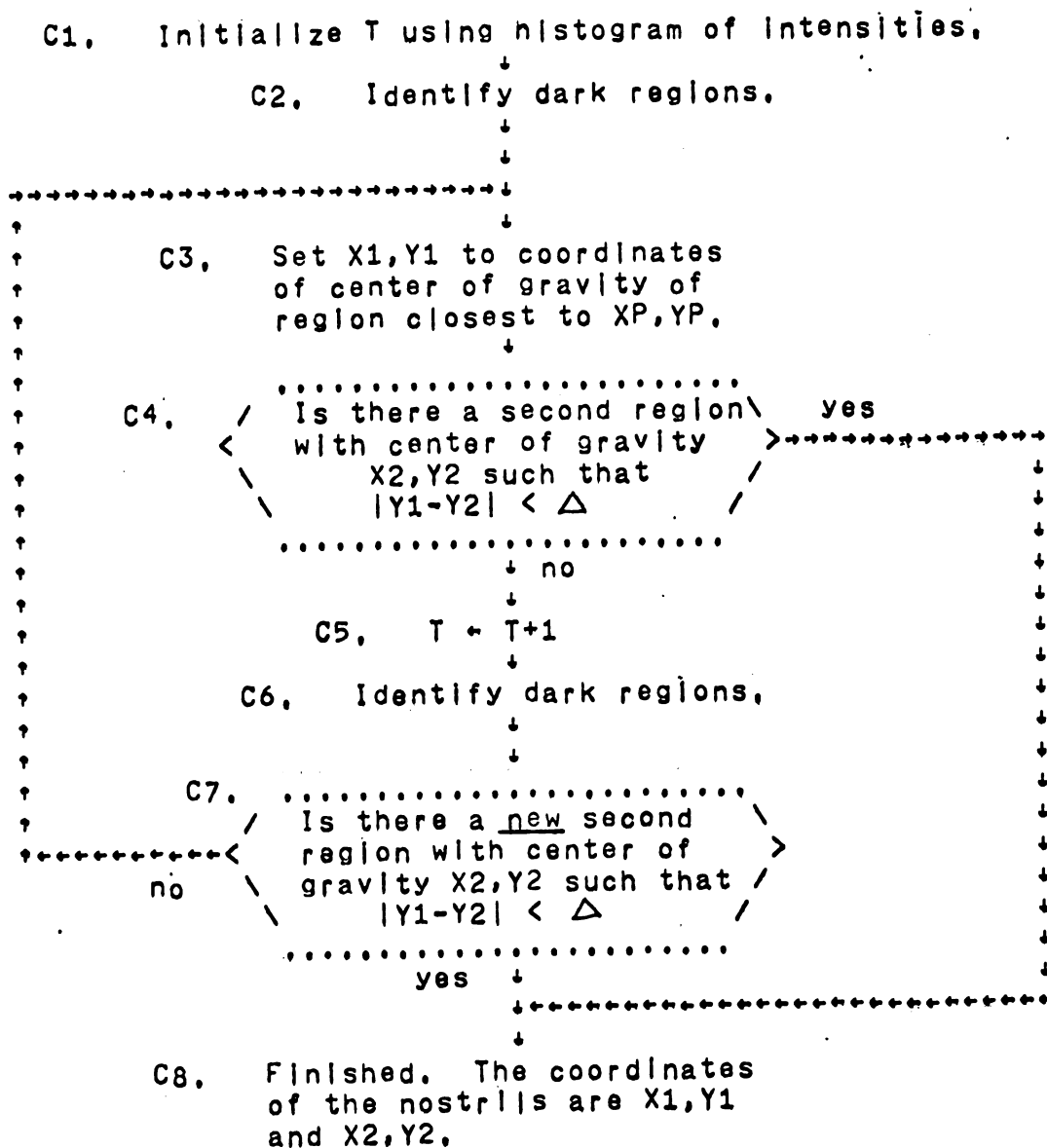


Figure 8-12. Locating the position of the nostrils.

region to which it belongs. From this information the centroid of each dark, connected region is computed.

C3, XP, YP are the coordinates of the point predicted to be between the nostrils. The closest region to XP, YP is found. Here the distance from a point to a region is defined as the Euclidean distance from the point to the centroid of the region. $X1$ and $Y1$ are set equal to the centroid of the closest region.

C4, A check is made to see if there is a second region in the picture at approximately the same height as $X1, Y1$. In particular, what is sought is a centroid $X2, Y2$ such that

$$|Y1 - Y2| < \lambda_1 \quad \text{and} \quad |X1 - X2| > \lambda_2,$$

Satisfactory values for the parameters are $\lambda_1=4$ and $\lambda_2=8$. If such a region is found, the search for the nostrils is successful; the program transfers to C8. If such a region is not found, more regions are searched for at C5.

C5. The threshold T is increased by one. This will include more dark points in the next iteration of the procedure.

C6, Dark regions are identified in the area of the nose using the new value of T . This step is identical with step C2.

C7, Hopefully, new regions have been found in the preceding step. The coordinates $X1, Y1$ of the old closest

region (found in step C3) are still known, A check is made to see if a new region is present at approximately the same height as $X1, Y1$. The details of the test are the same as in step C4. If there is a new region which meets this criteria, the search for the nostrils is successful, and the program continues to step C8. If not, another iteration of the procedure is needed and a transfer is made to step C3.

C8. The procedure is finished, $X1, Y1$ and $X2, Y2$ represent the coordinates of the two nostrils.

This completes the processing of the nose. The locations of the nostrils are recorded for eventual use in the classification step.

MOUTH.

The mouth was a difficult feature to locate consistently. When work was first begun, it was anticipated that points of the mouth and lips could be found by edge detection operators. This did not work because of the great variability present in the mouth region. In a light intensity representation of the mouth and its surroundings, the things that stand out are the highlights and shadows caused by light falling on the recesses and protruding curves of the mouth. The lips and the mouth itself are much less prominent. These lights and shadows change greatly with various lighting arrangements.

Another problem in recognizing mouths is the

variability of position. There is no "standard" position that a subject can be instructed to assume with his mouth. Therefore, pictures of the same mouth look different from picture to picture.

The one characteristic of the mouth that is uniformly detectable is the dark thin horizontal line which the mouth forms. The program was written to detect the presence of this line and its location.

The principal component of this process is a line-finding operator. A great deal of experimentation was performed in an attempt to devise a good operator. The following procedure is the result.

The operator detects horizontal dark lines. It examines points in a 7x7 square of the picture. Let B_i be the average intensity value in row i of the picture.

If

$$B_1 \geq B_2 \geq B_3 \geq B_4 \leq B_5 \leq B_6 \leq B_7$$

and if

$$B_1 > B_4 < B_7$$

then the point at the center of the square is on a dark line. If the two relations above do not hold, the central point is not on a dark line.

Now the procedure for locating the mouth can be described.

Step 1. The area to be searched for the mouth is delimited. This area extends vertically from below the nose

to near the predicted chin. Horizontally, it extends from left to right just outside of the previously located eye centers.

Step 2. The mouth region is smoothed using 2×2 smoothing. Here as elsewhere, smoothing improves performance.

Step 3. Within the mouth region all points that are on dark horizontal lines are identified. The 7×7 line finding operator described above is used.

Step 4. A threshold is determined such that half of the points which are on dark horizontal lines are darker than this threshold. Points with values above this threshold are eliminated.

Step 5. The rows of the region under consideration are examined to find that row which contains the greatest number of the points remaining. At this point the mouth is easily identified as the string of remaining dark points on adjacent rows.

This completes the processing of the mouth, the final feature on the face to be determined. All measurements are now available. The classification algorithm can be applied to the measurements to identify the individual.

CHAPTER 9

PATTERN CLASSIFICATION

This chapter will describe the methods used to identify a subject once a set of measurements has been obtained from the picture. It is worthwhile to re-emphasize that the study of pattern classification methods does not represent a major part of this thesis. The location and identification of features in grey scale pictures is the concern of this thesis. Once the features are located and measurements obtained, it is necessary to use a classification algorithm for identification of the subject. For this purpose a standard and easily used classification method is used.

The pattern classification problem is an example of decision-making when there is uncertainty in the data on which the decision is based. It may be defined as follows. Given a real valued vector $\vec{x}$, which has been selected from one of the classes $C_1, C_2, \dots, C_m$, we wish to find which of the classes that $\vec{x}$ represents. That is, we wish to find a decision function of the feature vector $\vec{x}$ such that

$$f(\vec{x}) = i \quad \text{if } x \in C_i.$$

In this thesis, the components of $\vec{x}$ are the measurements obtained from the picture, normalized by dividing each measurement by the measured height. The classes C_i each represent one person whose measurements are in the recognition dictionary.

This pattern classification problem may be separated from experimental pattern recognition and treated as a problem in mathematical statistics. There exists an extensive literature concerning this problem. Many classification algorithms have been described. Various algorithms are appropriate depending on the knowledge which is available about the probabilities of the classes and the probability distributions of the features for members of each class. The many methods which have been proposed and used are broadly surveyed and described in the recent papers by Nagy [1968] and Ho and Agrawala [1968]. Both papers contain lengthy bibliographies of the important literature dealing with pattern classification.

Identification of people, as Bledsoe and Hart have observed, is characterized by a small number of samples from each of a large number of classes. This contrasts with traditional pattern recognition problems where one has many samples from a few classes. Thus it is difficult to estimate probability distributions.

Accordingly, from the many classification schemes that are available, the nearest neighbor method (Cover and Hart [1967]) was chosen for this work in identification of people. The reason for this choice is that the nearest neighbor method is simple, easy to use, intuitively appealing, and requires no prior knowledge of underlying probability distributions.

The nearest neighbor method may be described as follows. For each person whose measurements are in the recognition dictionary, one or several n -component measurement vectors have been obtained. Each measurement vector may be considered as specifying a point in an n -dimensional space. A distance function $D(\vec{x}, \vec{y})$ is defined between any two points $\vec{x}$ and $\vec{y}$ in the space. When it is necessary to classify a new measurement vector $\vec{x}$, the distances between $\vec{x}$ and each point in the dictionary are computed. The closest point to $\vec{x}$ is that $\vec{y}_i$ for which $D(\vec{x}, \vec{y}_i)$ is minimum, and $\vec{x}$ is identified as representing the same person as $\vec{y}_i$.

With relatively few vectors for each person in the dictionary, this sort of classification will work well if the feature vectors for each person are nicely clustered in n -space. The success of the Bertillon system and Biedsoe's work with identification from measurements indicates that this is indeed the case for accurate measurements. In the recognition system studied here, the measurements contain measurement errors, but the nature of these errors is difficult to estimate. Nevertheless, the nearest neighbor classification method provides a simple way to test the feature detection that forms the principal effort in this thesis.

As stated before, the nearest neighbor procedure is simple and intuitively appealing. Its chief disadvantage,

the large memory required to store every previously encountered measurement, is not bothersome with the relatively small data sets used here. There is another feature of the nearest neighbor procedure which is worth mentioning. It is well known that when all the underlying probability structure of a classification problem is known and used, the optimum decision rule may be obtained using Bayes analysis and choosing the class C_i which maximizes

$$P(C_i | \vec{x}) = \frac{P(\vec{x} | C_i) P(C_i)}{\sum_{j=1}^m P(\vec{x} | C_j) P(C_j)},$$

Cover and Hart [1967] have shown that for large samples, the probability of error of the nearest neighbor rule is bounded above by twice the Bayes probability of error. (Minsky and Papert [1969, p. 197] give a clear example of a special case of this result.)

In choosing the distance function for use in the nearest neighbor algorithm, the metric which was used successfully by Bledsoe [1966] was selected. The distance from $\vec{x}$ to $\vec{y}$ is defined as

$$D(\vec{x}, \vec{y}) = \sum_{j=1}^n \frac{1}{\sigma_j^2} (x_j - y_j)^2$$

where σ_j^2 is the intra-person difference variance for measurement j ; in other words, σ_j^2 is the variance of differences in measurement j between pictures of the same person. More precisely, let $\vec{a}$ and $\vec{b}$ be two measurement vectors for person i . Form the difference vector $\vec{a} - \vec{b}$

for person i as

$$a_{ij} = a_j - b_j.$$

Do the same for each person until a complete set of difference vectors $\vec{a_1}, \vec{a_2}, \dots, \vec{a_m}$ has been obtained. Define σ_j^2 as the variance of the j th component of these difference vectors:

$$\sigma_j^2 = \frac{1}{m} \sum_{i=1}^m (a_{ij})^2 - \left[\frac{1}{m} \sum_{i=1}^m a_{ij} \right]^2.$$

Clearly this distance measure is a Euclidean distance on a space in which each component has been weighted by a confidence level $\frac{1}{\sigma_j^2}$. This weight will be large on a measurement that is reliably repeatable. It will be small on an unreliable measurement.

It was mentioned above that this is the distance measure used by Bledsoe. To be precise, it should be mentioned that Bledsoe had many more measurements in his manually obtained data. These measurements were grouped according to the features. For instance, one group might be all the measurements on points of the mouth. The distance measure presented above is the measure used by Bledsoe for the distance between centroids of groups.

CHAPTER 10

RESULTS AND CONCLUSIONS

The principal positive result of this research is the use of goal-directed techniques to successfully locate features in cluttered digital pictures. This success has been verified by displaying the results of the feature finding algorithms and comparing these locations with the locations obtained by hand from digital printouts of the pictures. Successful performance in the task of identification of people provides further verification for the feature finding algorithms. The test of the performance of the program on identification of people is described below,

A collection of 72 digital pictures was used in this test. This collection comprised 24 sets of pictures. Each set consisted of 3 pictures for one individual: a picture of the body, a picture of the background which corresponded to the body picture, and a close-up of the head. Ten individuals were represented in the collection. For most, there were 2 sets of pictures. Three individuals had 3 or 4 pictures each in the collection. The pictures of the same individual were taken at various times over a two year period.

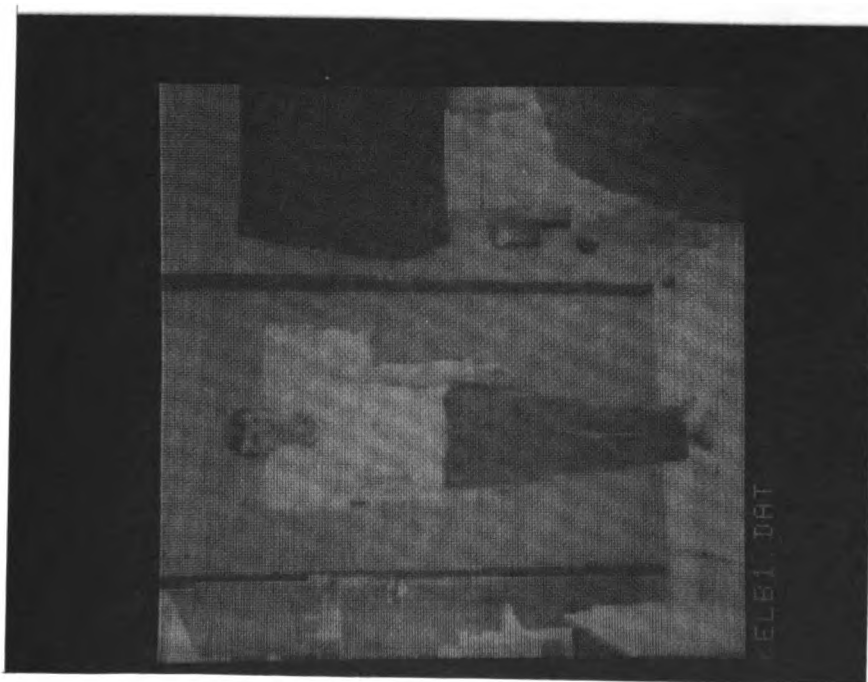
A large amount of information will be included in this chapter about the collection of pictures which was used in

the performance test. This is done in order to provide a clear indication of the quality of the input and the characteristics of the results. In addition this information could be used in comparison of alternate methods with the methods of this thesis. A full set of the data used to test this program is available from the author.

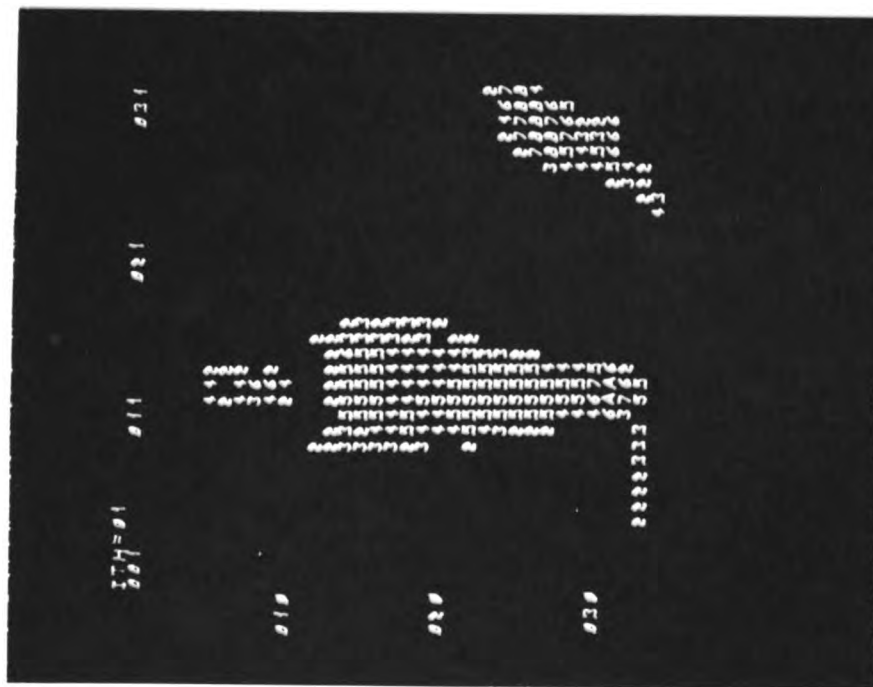
Detailed information on the pictures of the entire body will be provided for 4 pictures, 2 pictures each of 2 individuals.

The complete input data for 2 of these pictures is included as Appendix 1. For each picture several pages of computer printout are given. Taken together these pages completely specify the input picture. In the printouts in Appendix 1 (and also in Appendix 2, to be described later) the sixteen levels of light intensity are represented as follows. Each point contains a light intensity value which ranges from zero (the darkest points) to fifteen (the lightest points). Intensity values 0 through 9 are represented by the digits "0" through "9". Intensity value 10 is represented by the letter "A"; 11 by "B"; and 12, 13, and 14 by the letters "C", "D", and "E" respectively. Intensity value 15 is printed as a period.

Each of Figures 10-1 through 10-4 gives 4 photographs derived from the 4 example pictures of bodies. Part (a) gives a grey scale representation which was obtained by photographing a computer display terminal. Part (b) shows

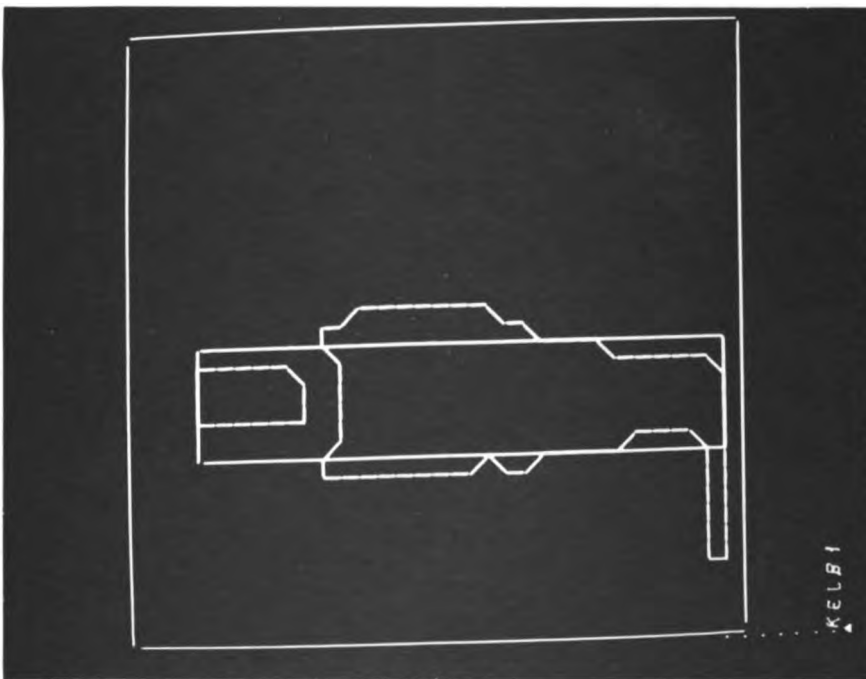


(a)

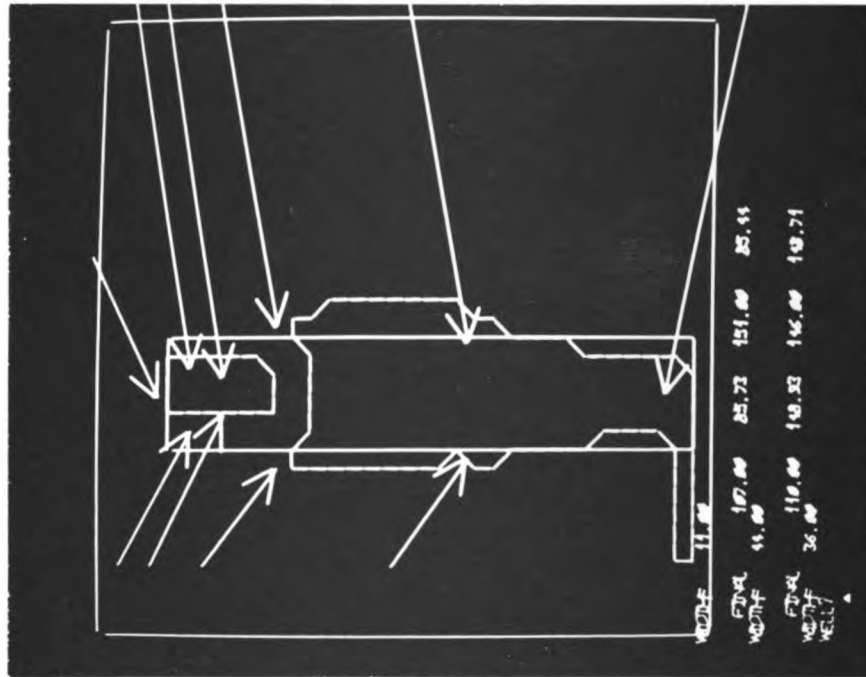


(b)

Figure 10-1. Picture KELBI

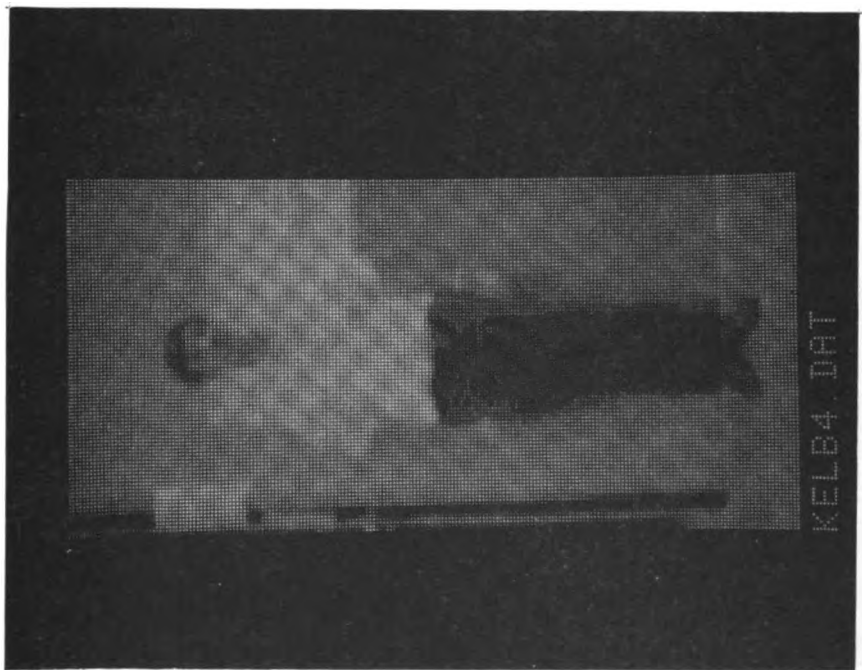


(c)

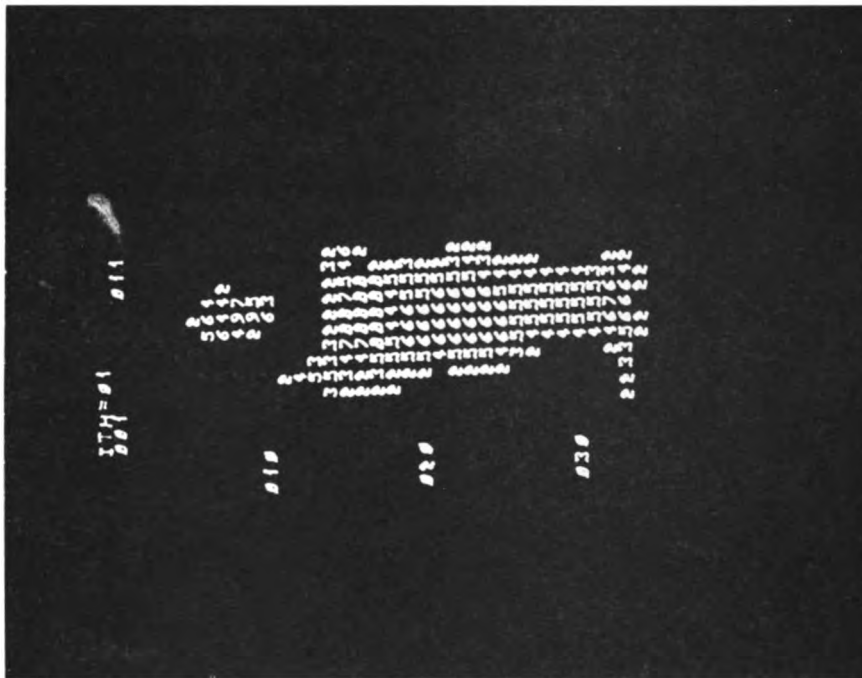


(d)

Figure 10-1 (continued)

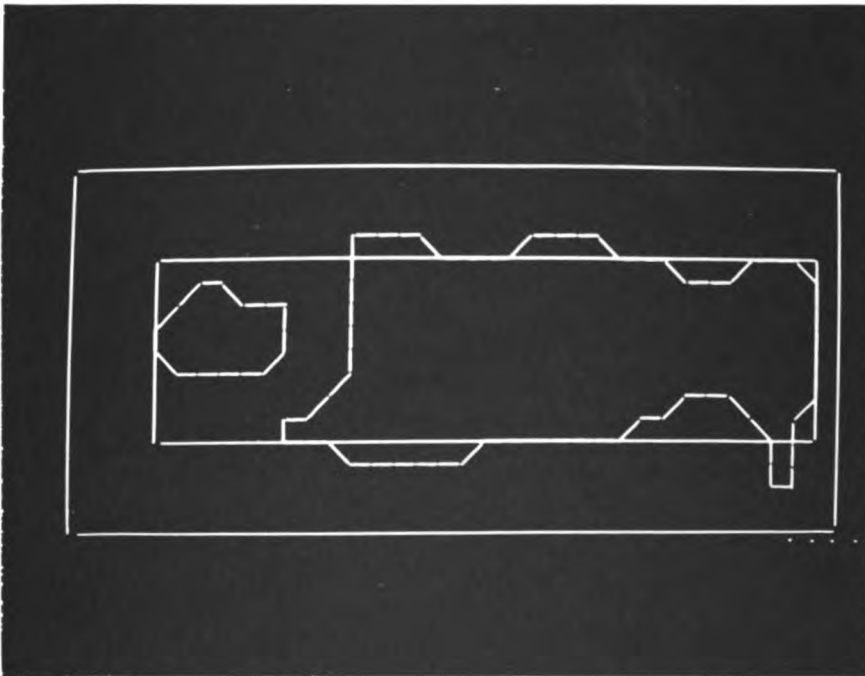


(a)

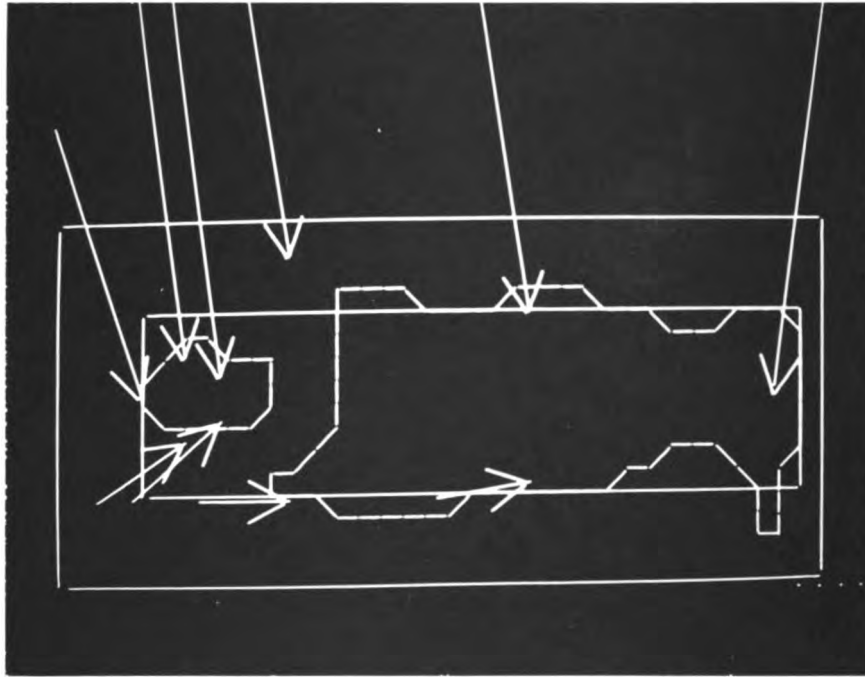


(b)

Figure 10-2. Picture KELB4



(c)



(d)

Figure 10-2 (continued)

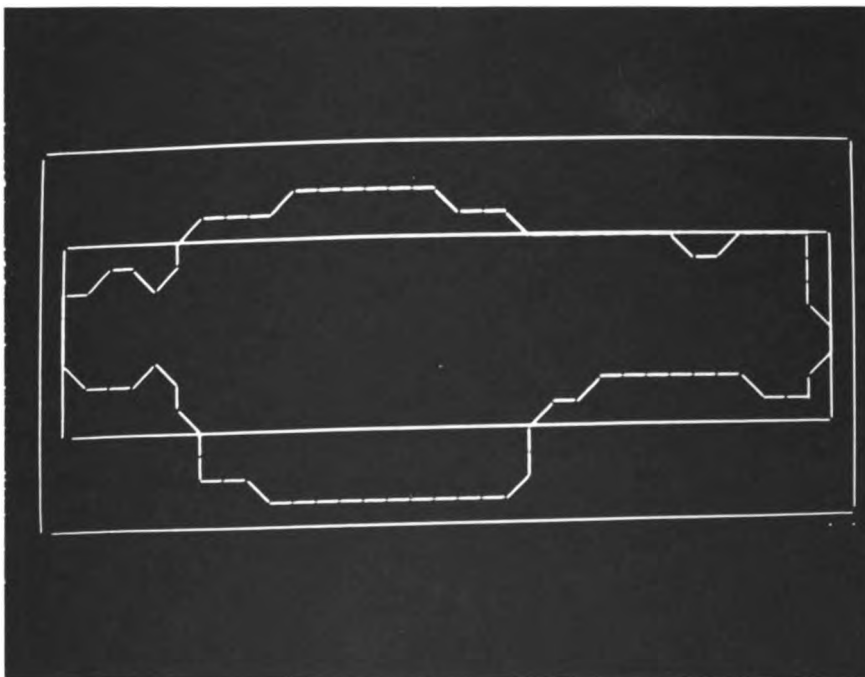


(a)

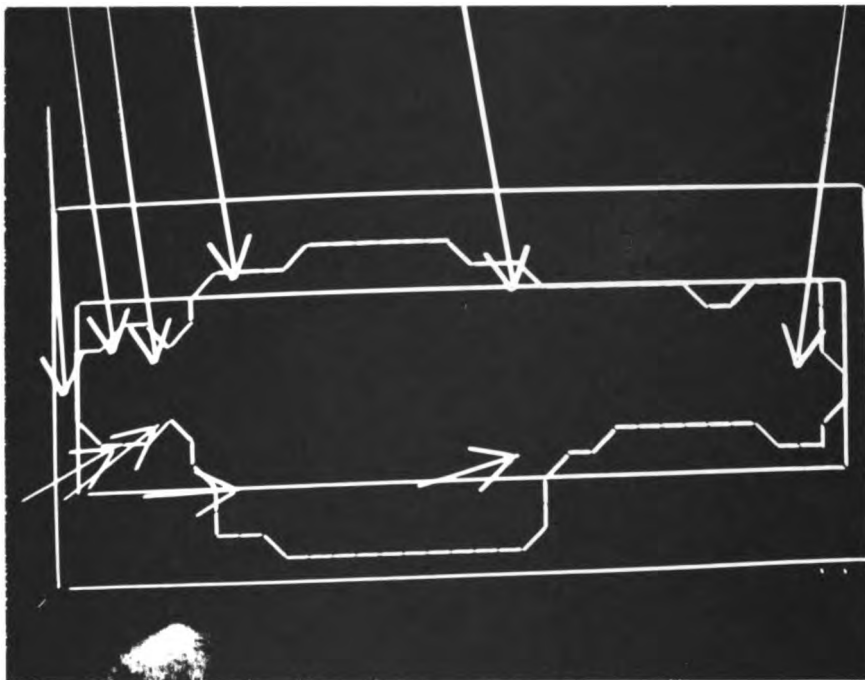


(b)

Figure 10-3. Picture JZCB1



(c)



(d)

Figure 10-3 (continued)

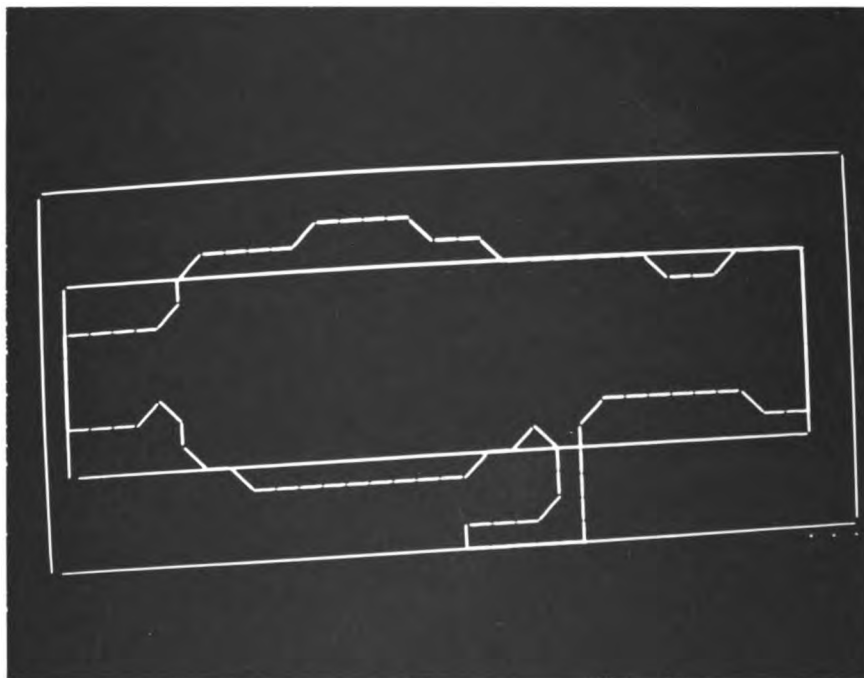


(a)

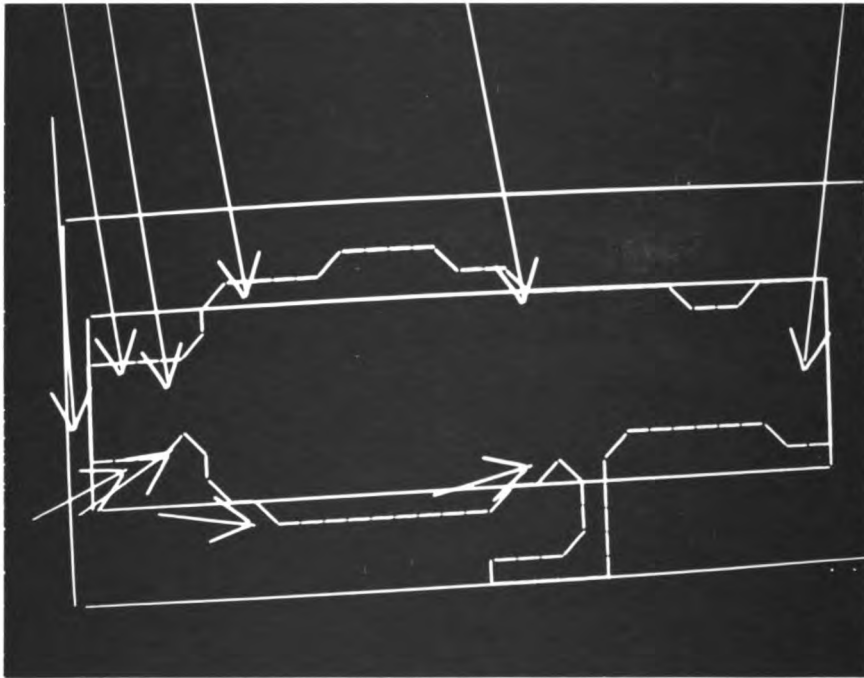


(b)

Figure 10-4. Picture JZCB2



(c)



(d)

Figure 10-4 (continued)

the results of processing the pictures after subtraction and reduction in size. The representation is the same as that used in Appendix 1 which was described above. Part (c) shows the outlines of the regions which remain after smoothing, noise elimination, and removal of the border regions. Part (d) shows the same outlines as part (c) with arrows superimposed which point to the locations of the measurement points.

Table 10-1 gives the coordinates of all of the measurement points located in the 4 example body pictures.

Appendix 2 consists of complete input data for 2 pictures of heads, both representing the same individual. The data representation is the same as that used in Appendix 1 and described above.

Figures 10-6 through 10-29 contain pictures of each of the 24 heads in the test collection. In each figure part (a) is a grey scale photograph from a computer display terminal. Part (b) shows the outline of the head, the features found, and coordinates of features located. The eyes are represented by an outline of the dark areas around the eyes. Similar outlines are given for the nostrils. The mouth is indicated roughly by horizontal lines. Around the nose and mouth some noise points have not been suppressed. On the right of the outline of the head the feature coordinates that are used in subsequent measurements are given. Figure 10-5 explains which numbers belong with

TABLE 10-1,
Coordinate measurements from 4 body pictures.

Picture KELB1

			x coord,	y coord,
Top of Head			129	45
Feet			130	213
Sides of head	left		119	56
	right		138	56
		Width=	19	
Sides of neck	left		124	67
	right		135	67
		Width=	11	
Sides of shoulders	left		107	86
	right		151	85
		Width=	44	
Sides of hips	left		110	149
	right		146	149
		Width=	36	

Picture KELB4

			x coord,	y coord,
Top of Head			143	46
Feet			142	219
Sides of head	left		132	58
	right		153	58
		Width=	21	
Sides of neck	left		137	68
	right		148	68
		Width=	11	
Sides of shoulders	left		116	88
	right		179	88
		Width=	63	
Sides of hips	left		120	153
	right		164	153
		Width=	44	

TABLE 10-1. (Continued)

Coordinate measurements from 4 body pictures.

Picture JZCB1

			x coord.	y coord.
Top of Head			142	25
Feet			143	214
Sides of head	left		129	38
	right		152	38
		Width=	23	
Sides of neck	left		134	49
	right		149	49
		Width=	15	
Sides of shoulders	left		117	71
	right		169	71
		Width=	52	
Sides of hips	left		123	142
	right		164	142
		Width=	41	

Picture JZCB2

			x coord.	y coord.
Top of Head			137	24
Feet			142	214
Sides of head	left		127	37
	right		150	37
		Width=	23	
Sides of neck	left		131	49
	right		146	48
		Width=	15	
Sides of shoulders	left		112	71
	right		167	69
		Width=	55	
Sides of hips	left		122	142
	right		162	141
		Width=	40	

Figure 10-5,
Explanation of measurements in
Figures 10-6 through 10-29,

On the right side of each figure are ten values, The meaning of these values is defined by their position as follows:

<NAME OF PICTURE>

<TOP OF HEAD, Y COORD,>

<HEAD, LEFT SIDE X COORD,>

<HEAD, RIGHT SIDE X COORD,>

<LEFT EYE X COORD,>

<LEFT EYE Y COORD,>

<RIGHT EYE X COORD,>

<RIGHT EYE Y COORD,>

<NOSTRILS, Y COORD,>

<MOUTH, Y COORD,>

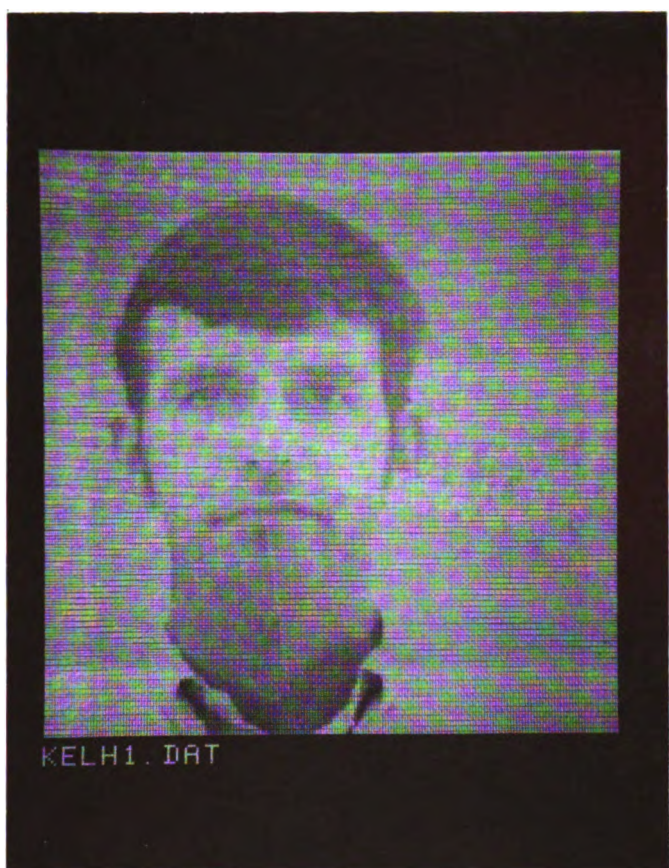
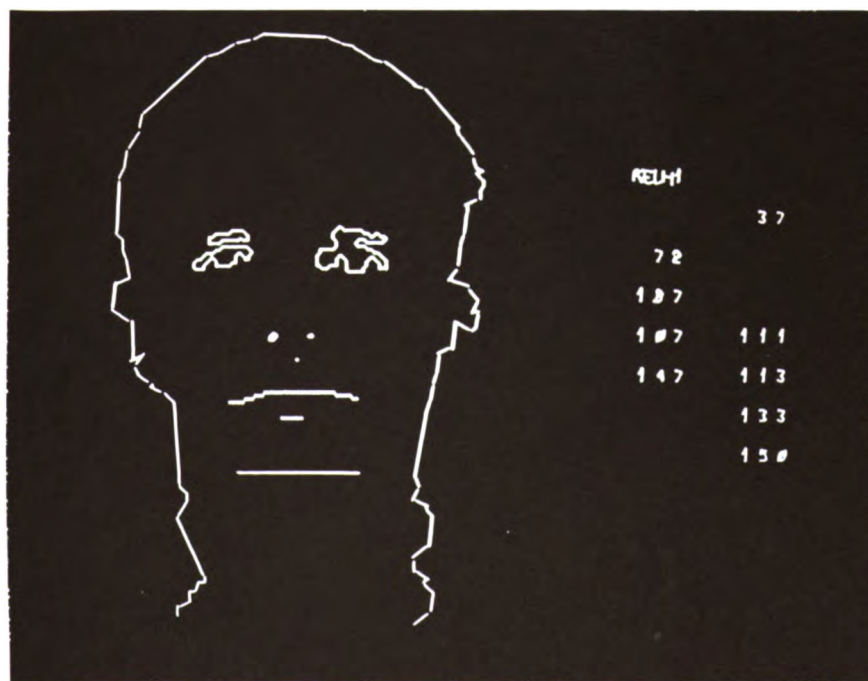


Figure 10-6

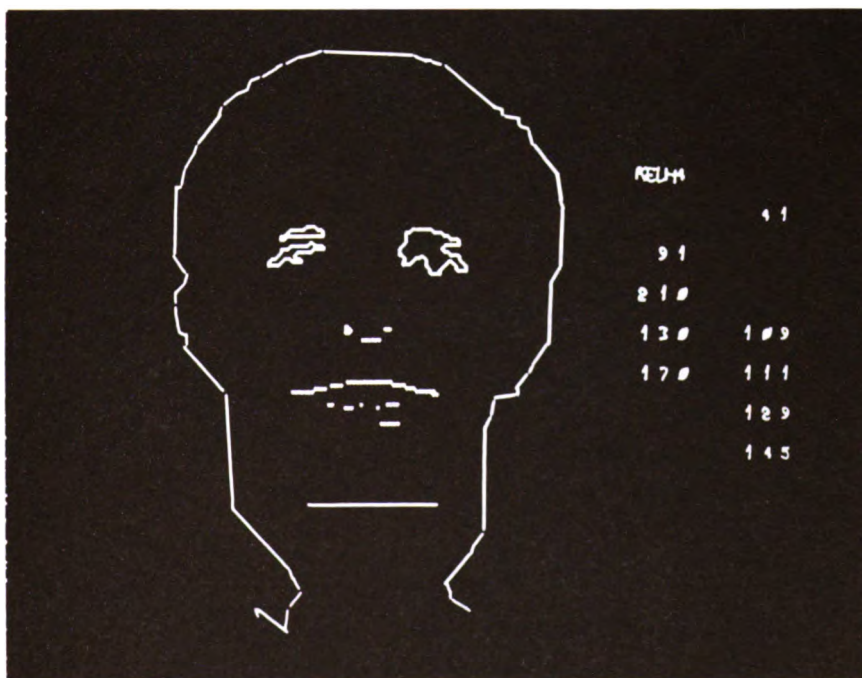


(b)



(a)

Figure 10-7

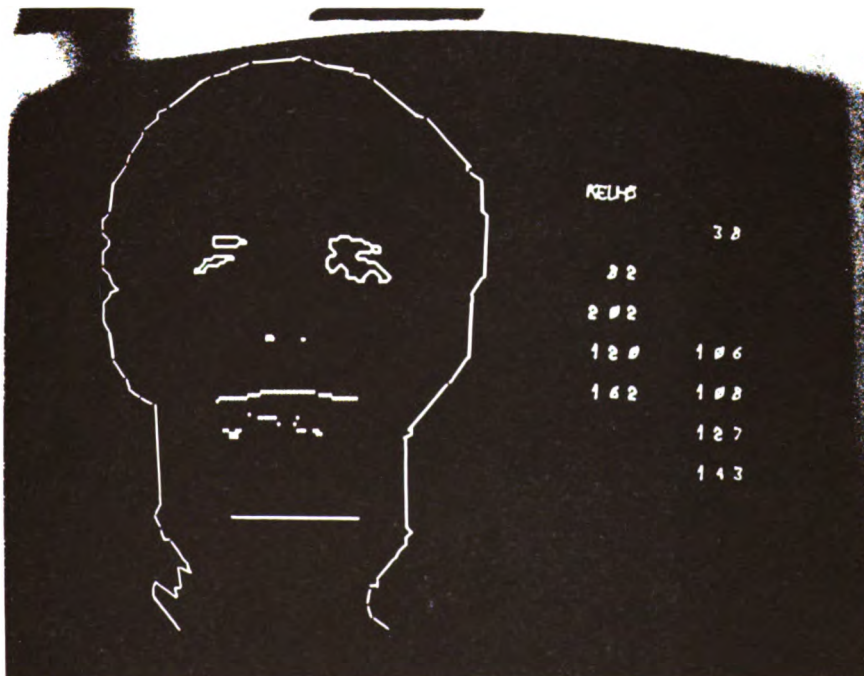


(b)

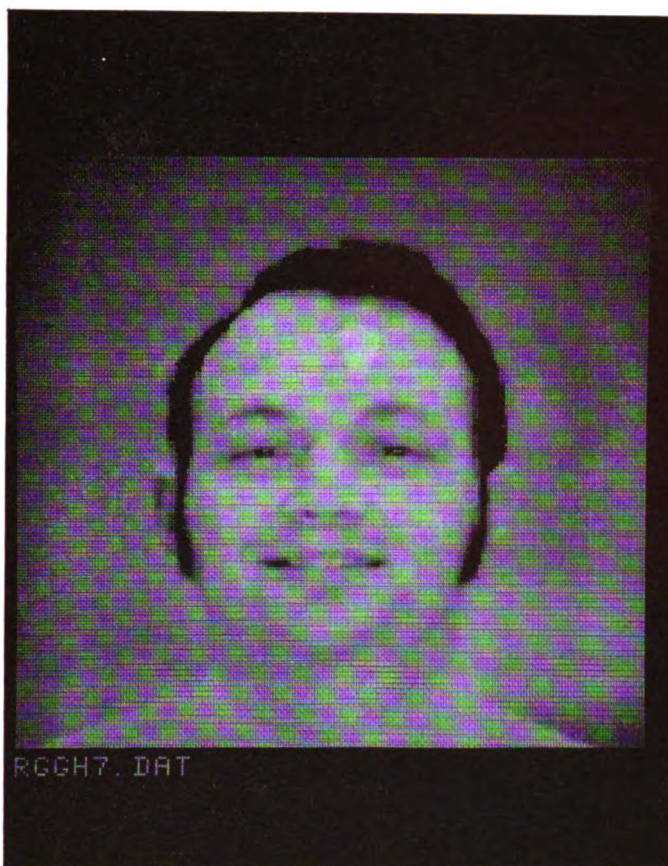


(a)

Figure 10-8

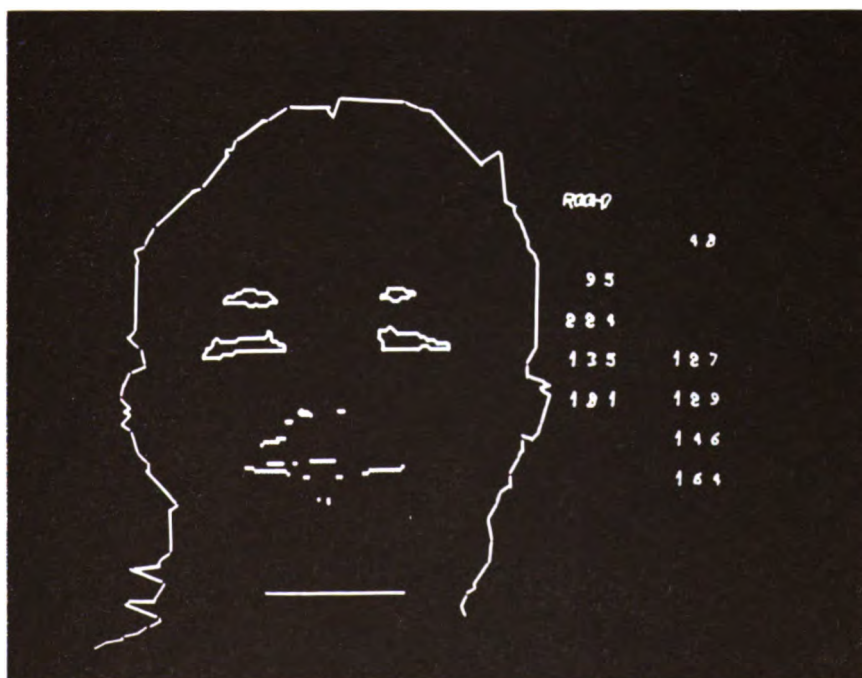


(b)



(a)

Figure 10-9

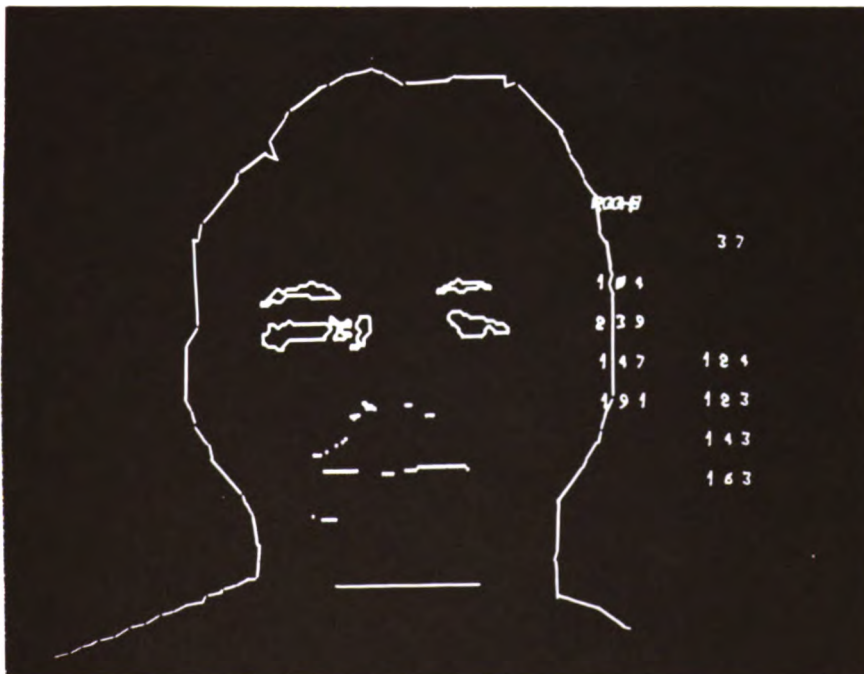


(b)



(a)

Figure 10-10

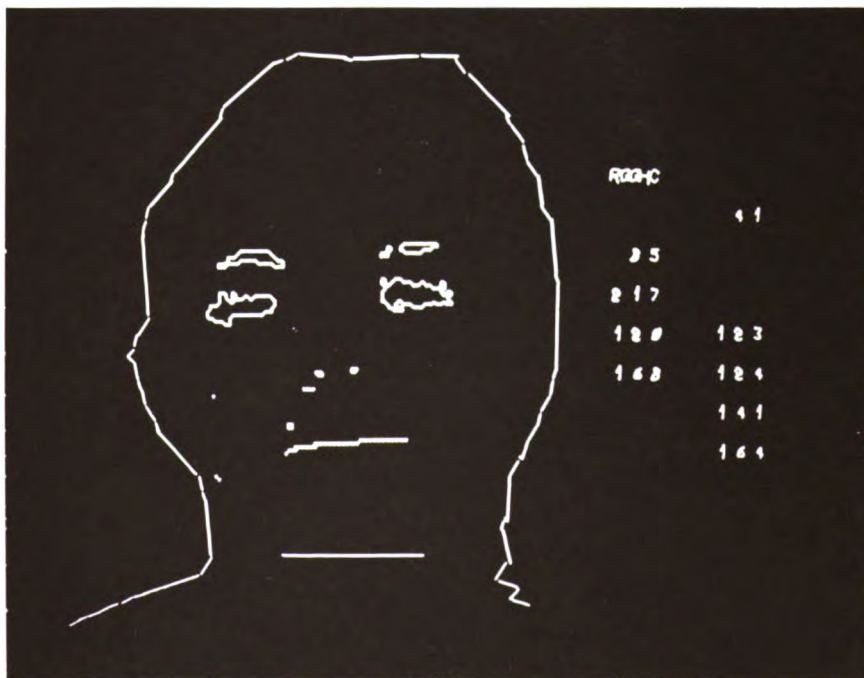


(b)



(a)

Figure 10-11

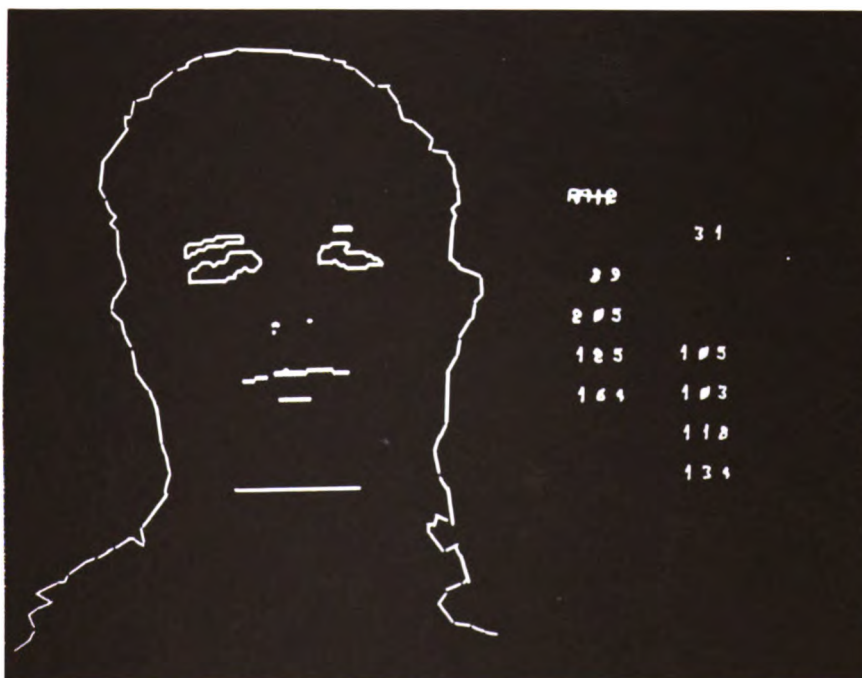


(b)



(a)

Figure 10-12

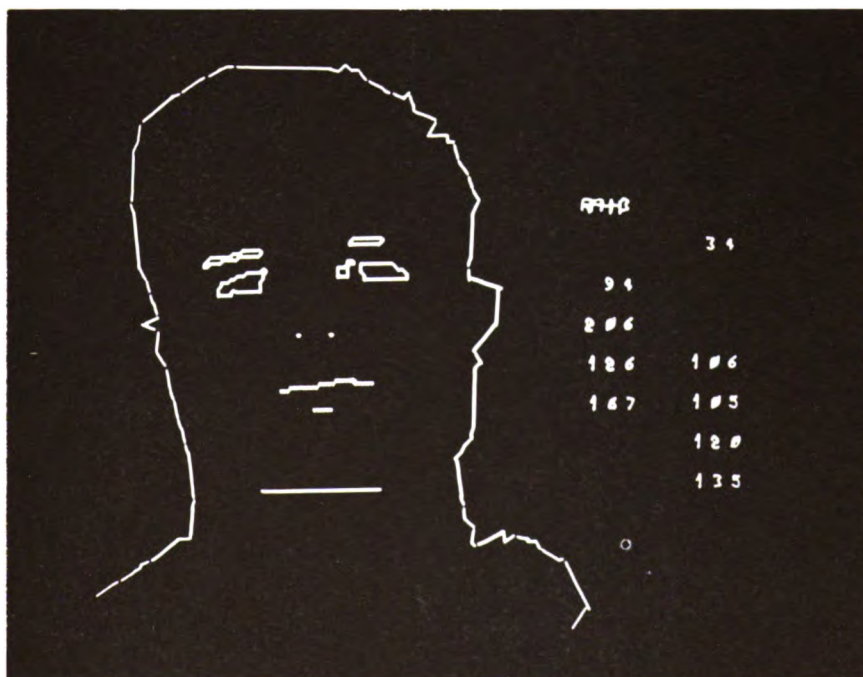


(b)

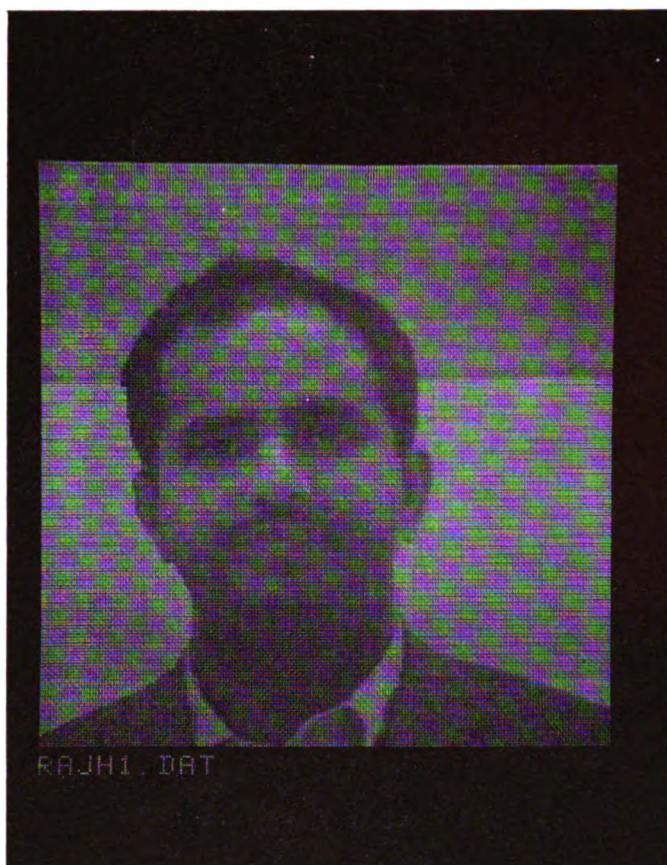


(a)

Figure 10-13

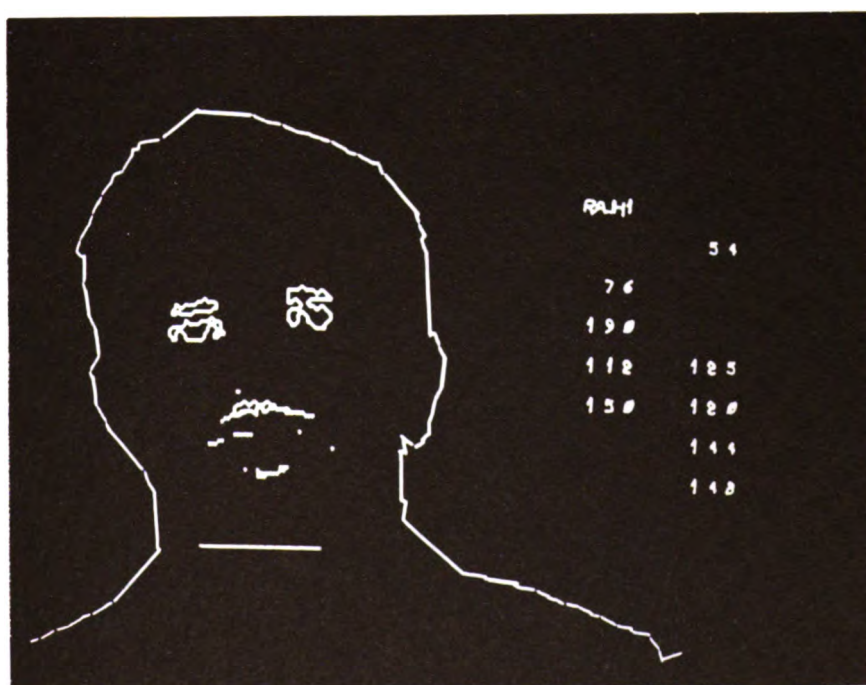


(b)

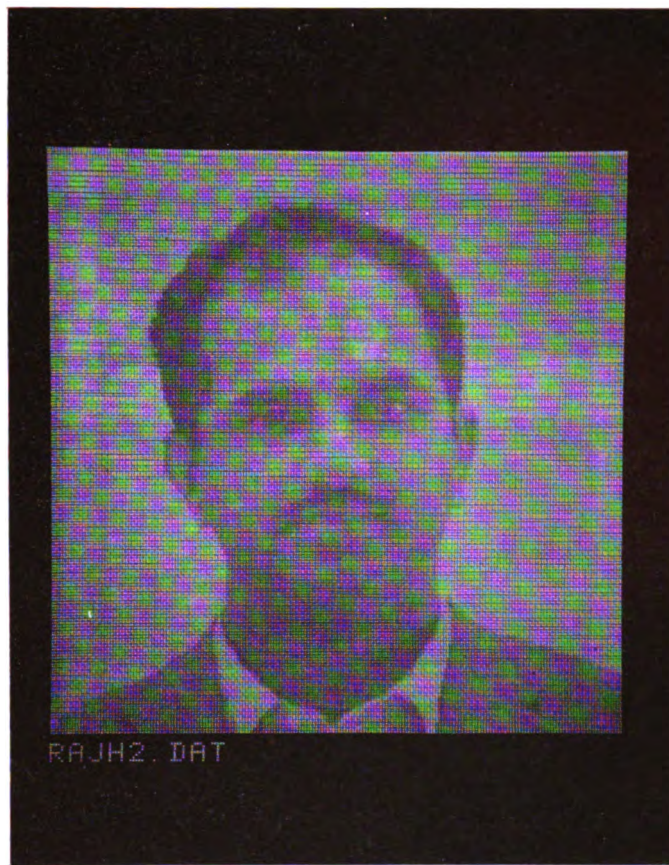


(a)

Figure 10-14

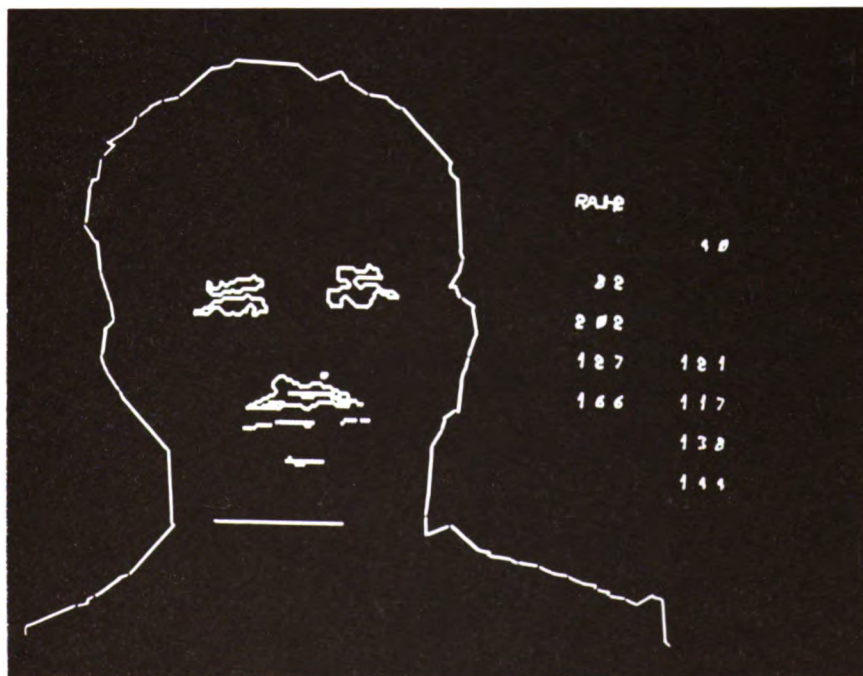


(b)

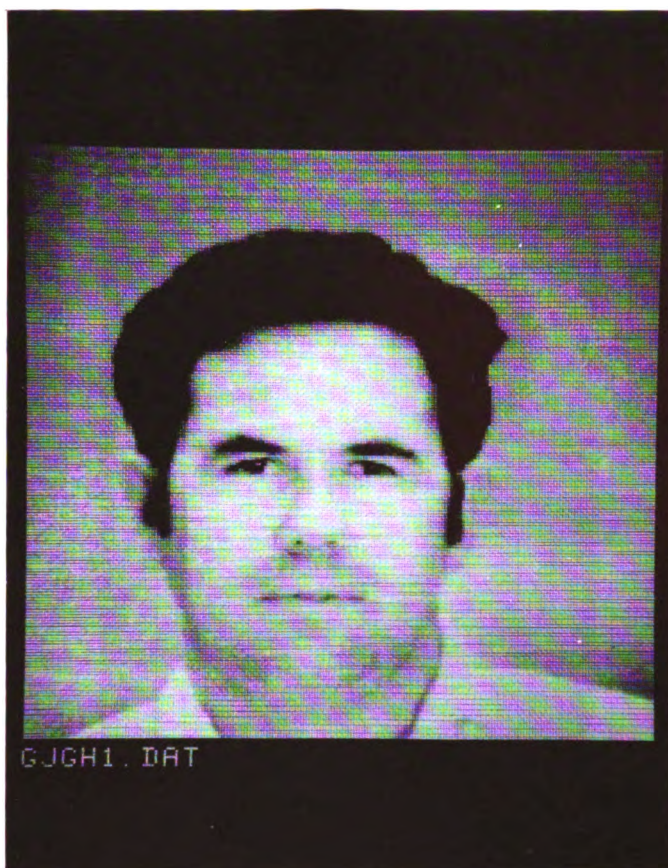


(a)

Figure 10-15

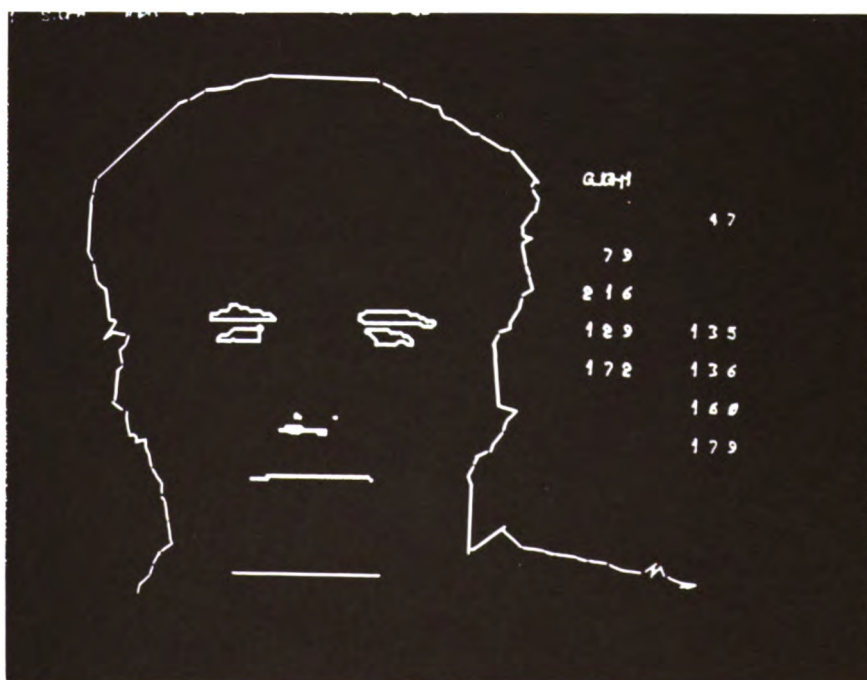


(b)

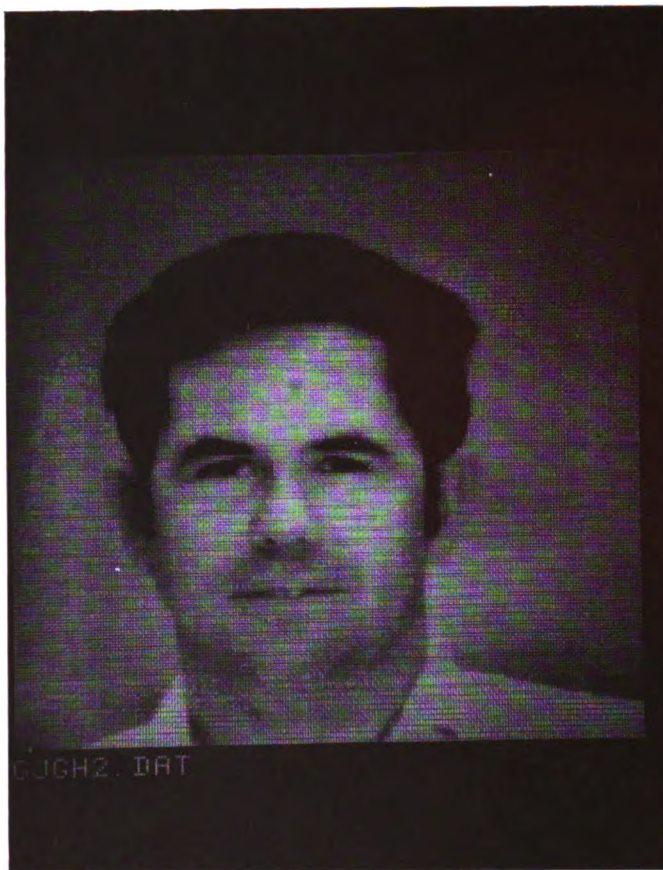


(a)

Figure 10-16

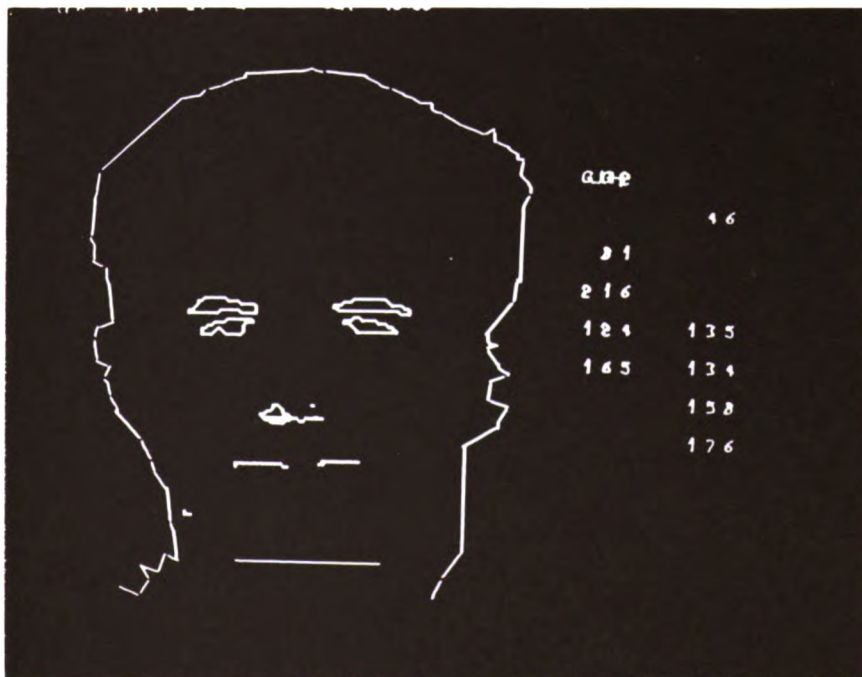


(b)

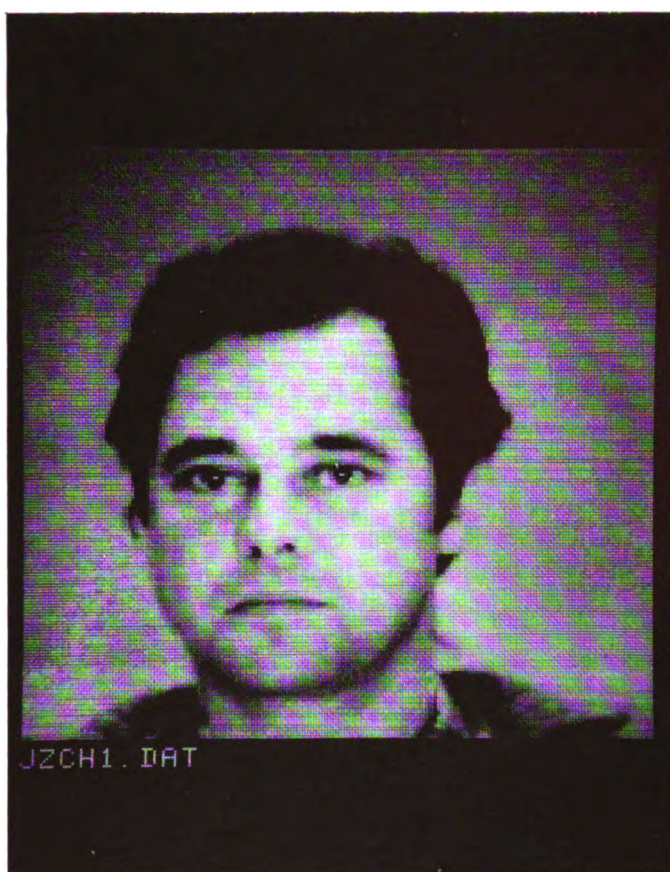


(a)

Figure 10-17

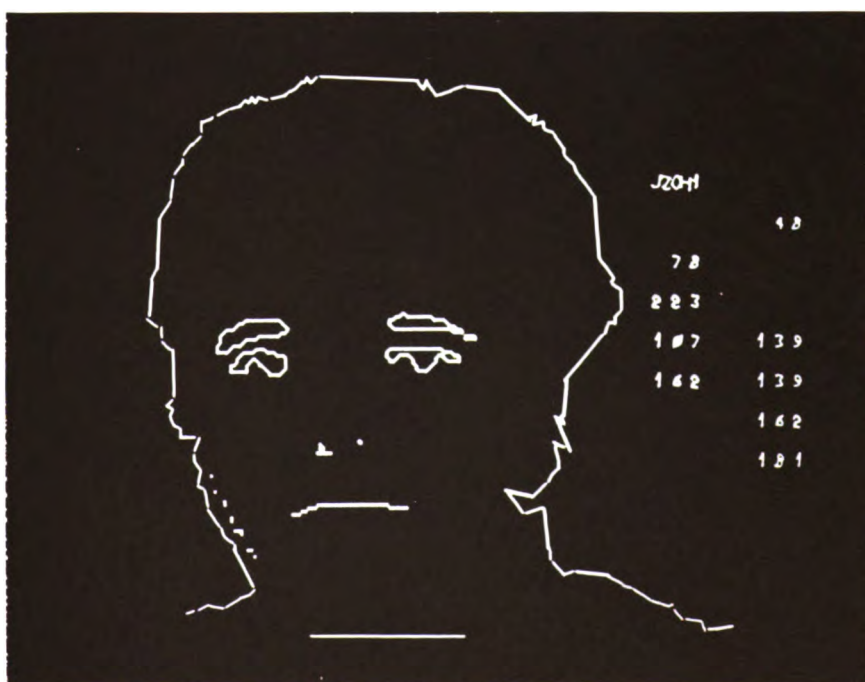


(b)

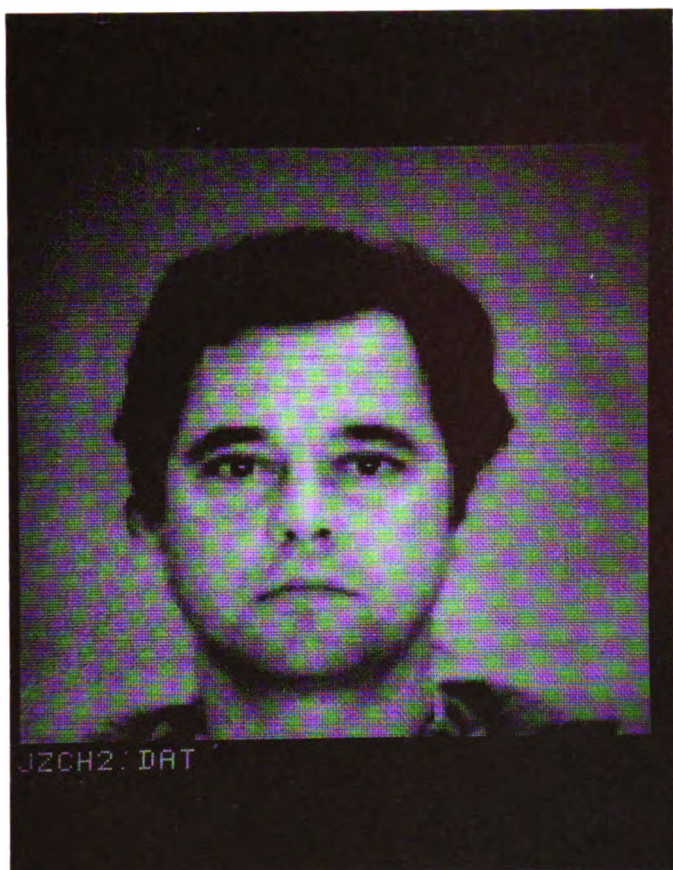


(a)

Figure 10-18

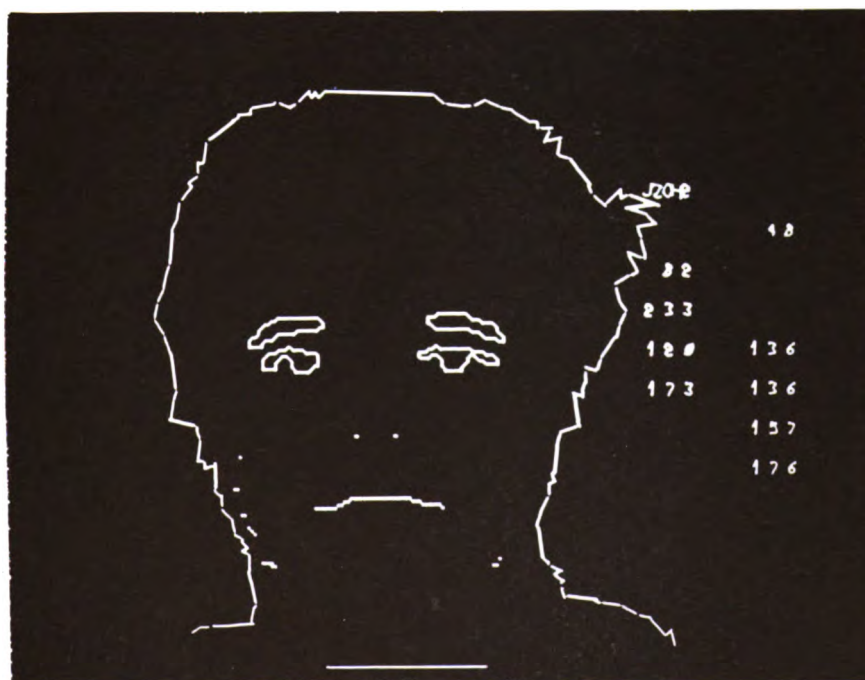


(b)

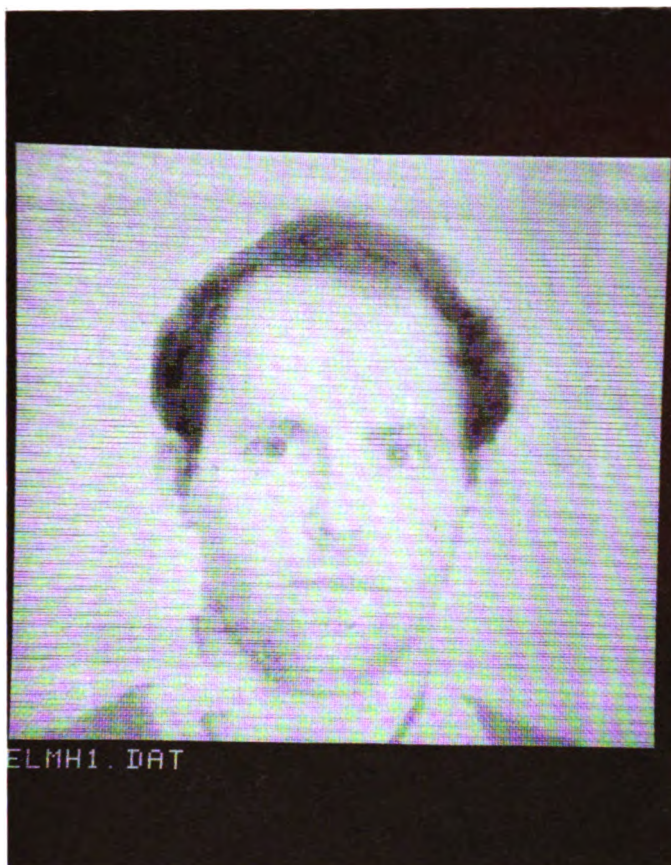


(a)

Figure 10-19

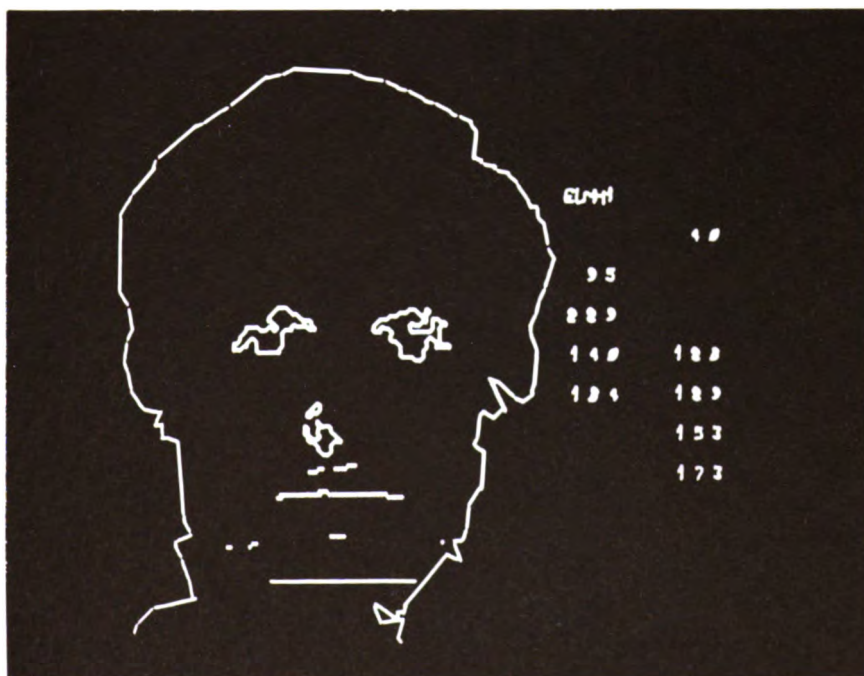


(b)

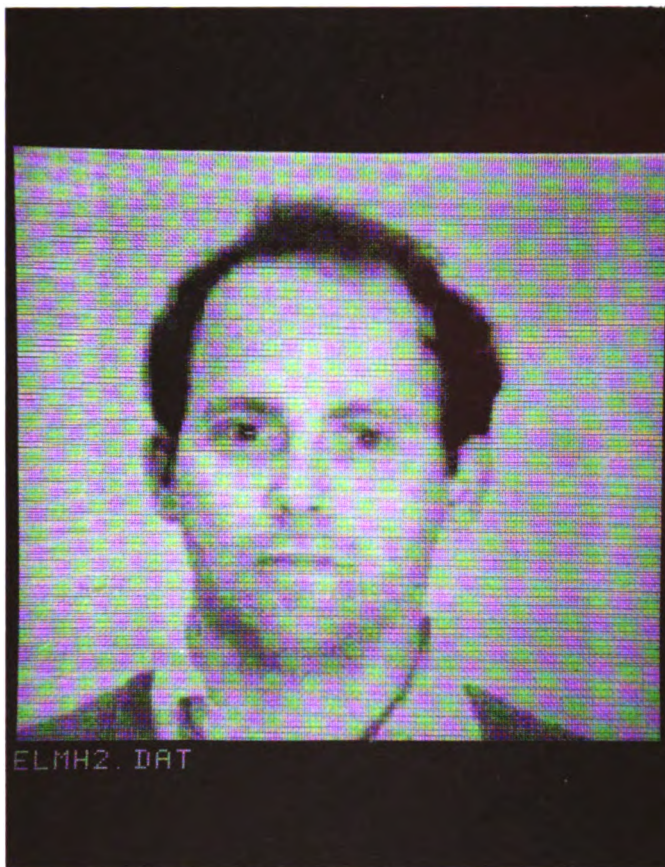


(a)

Figure 10-20

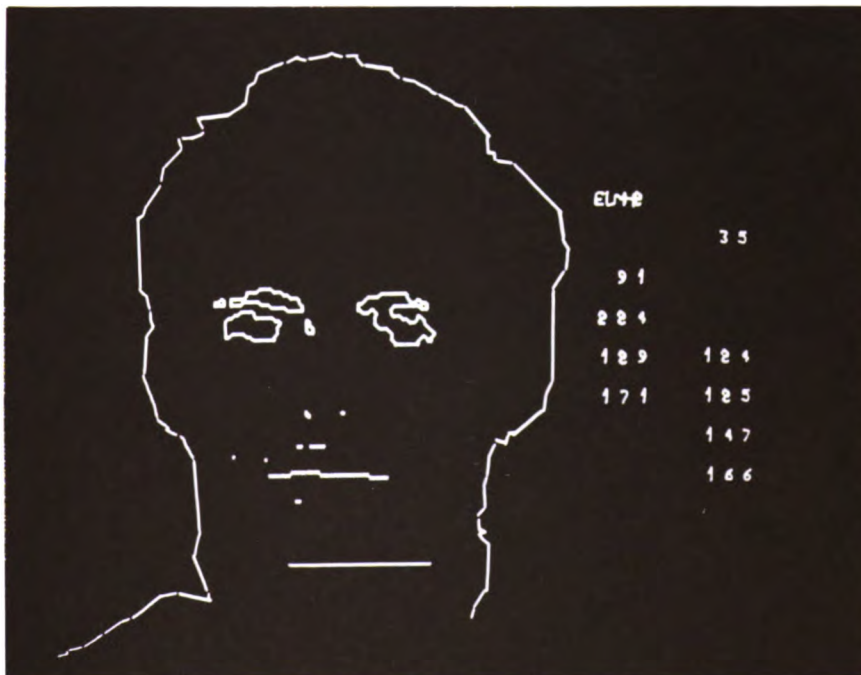


(b)



(a)

Figure 10-21

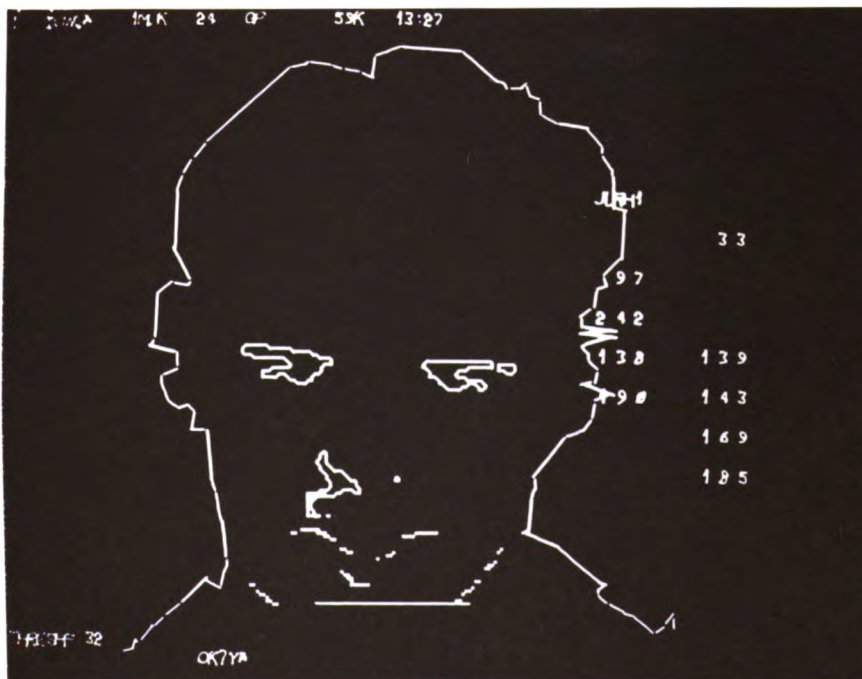


(b)

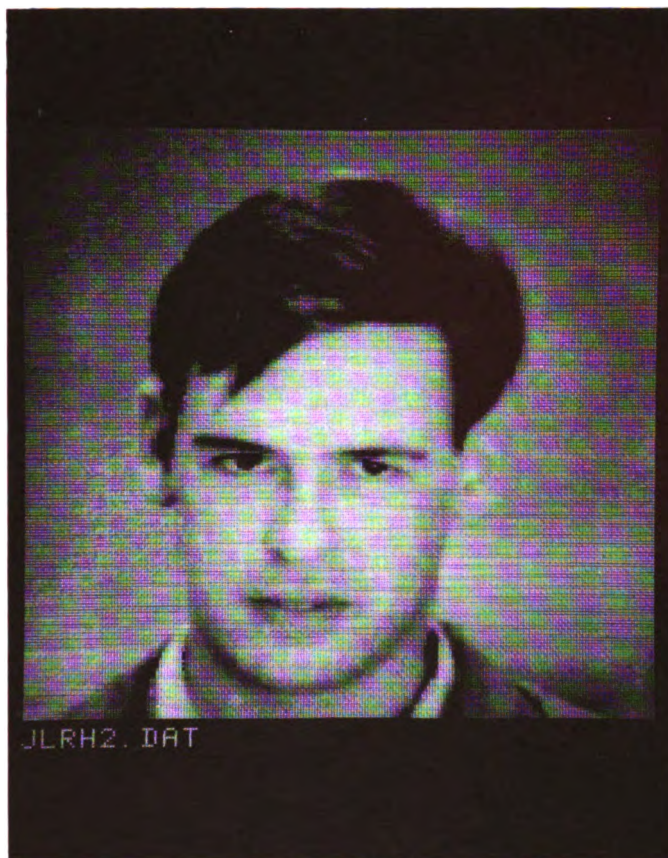


(a)

Figure 10-22

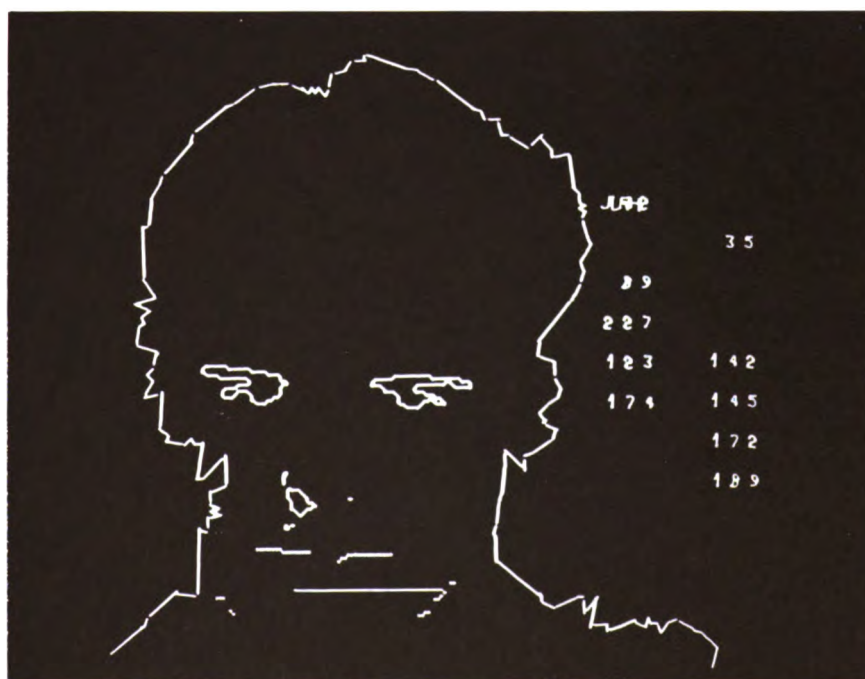


(b)



(a)

Figure 10-23

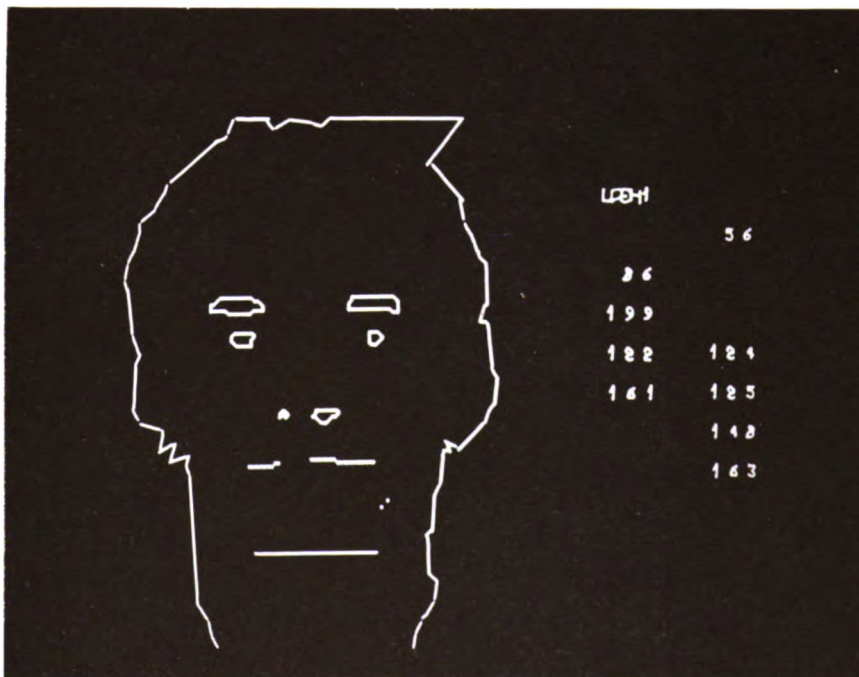


(b)



(a)

Figure 10-24

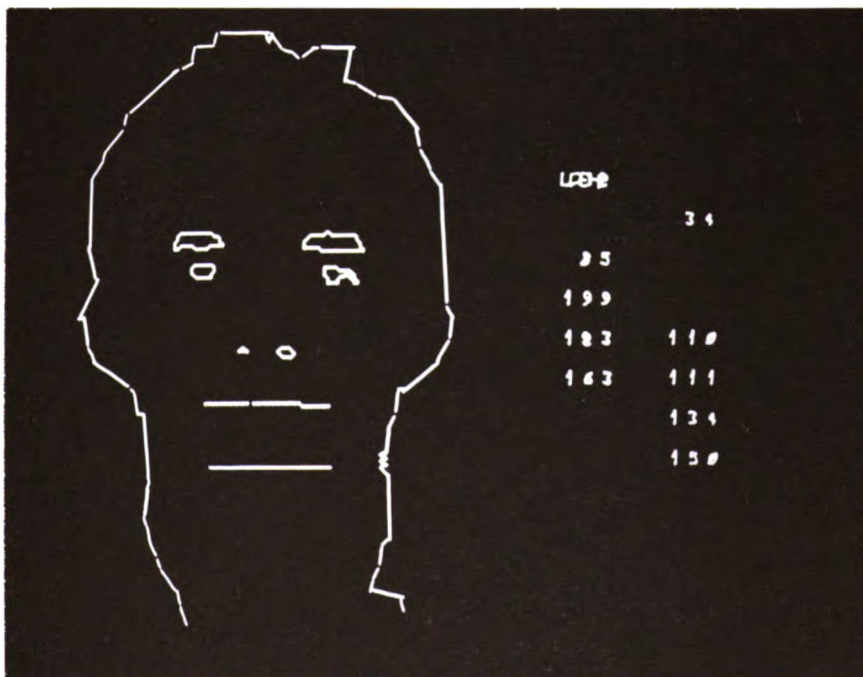


(b)



(a)

Figure 10-25

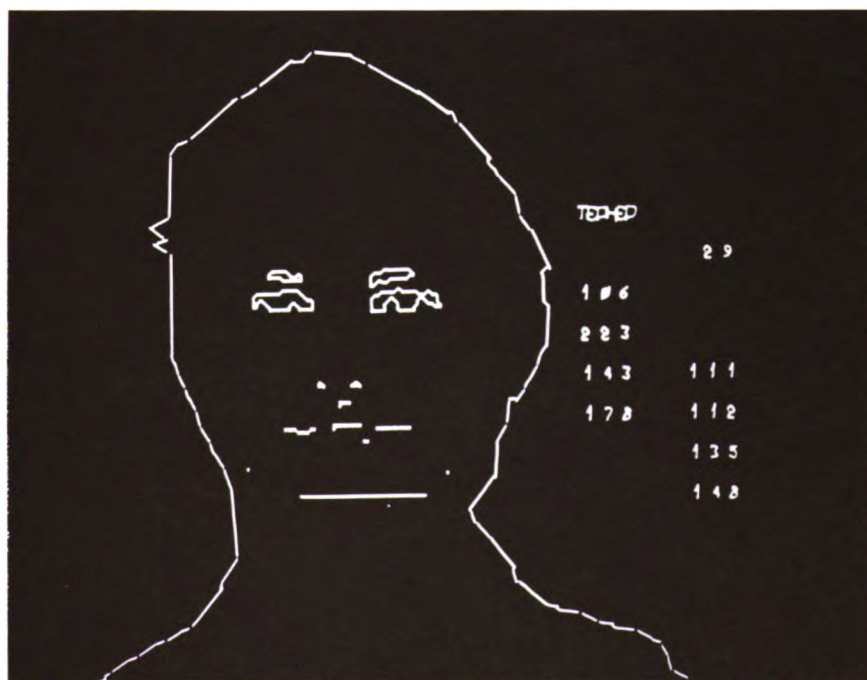


(b)

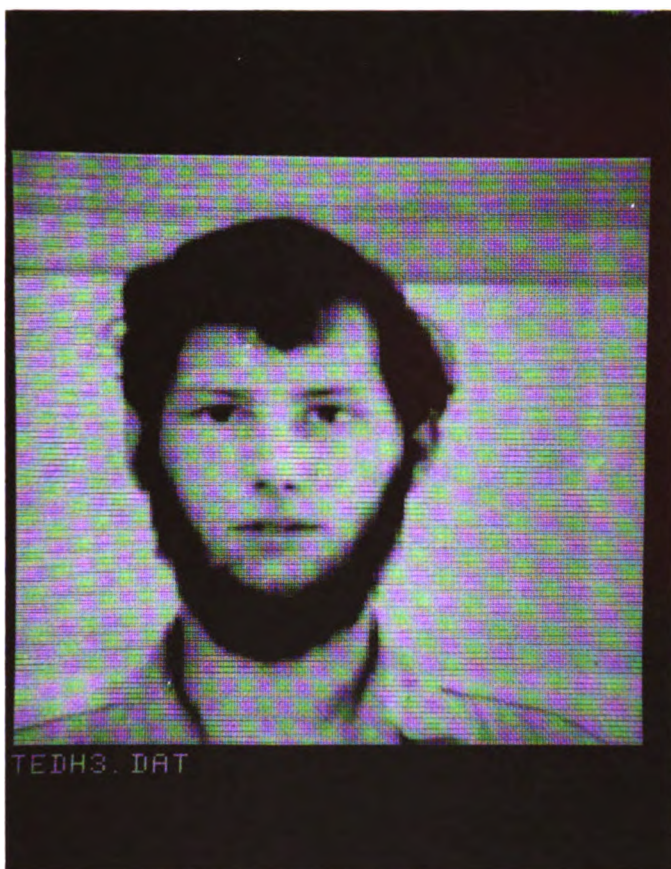


(a)

Figure 10-26

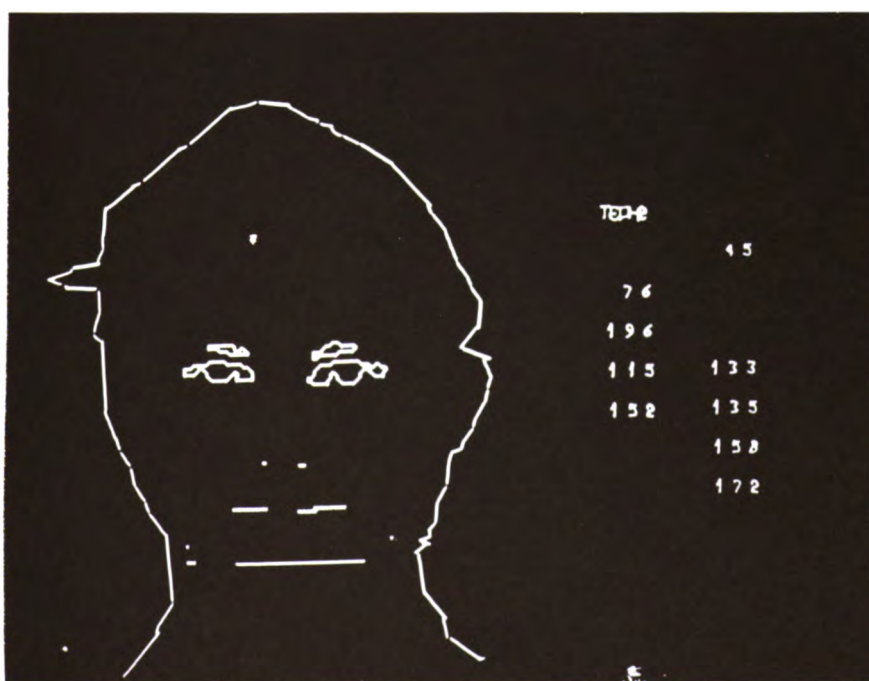


(b)



(a)

Figure 10-27

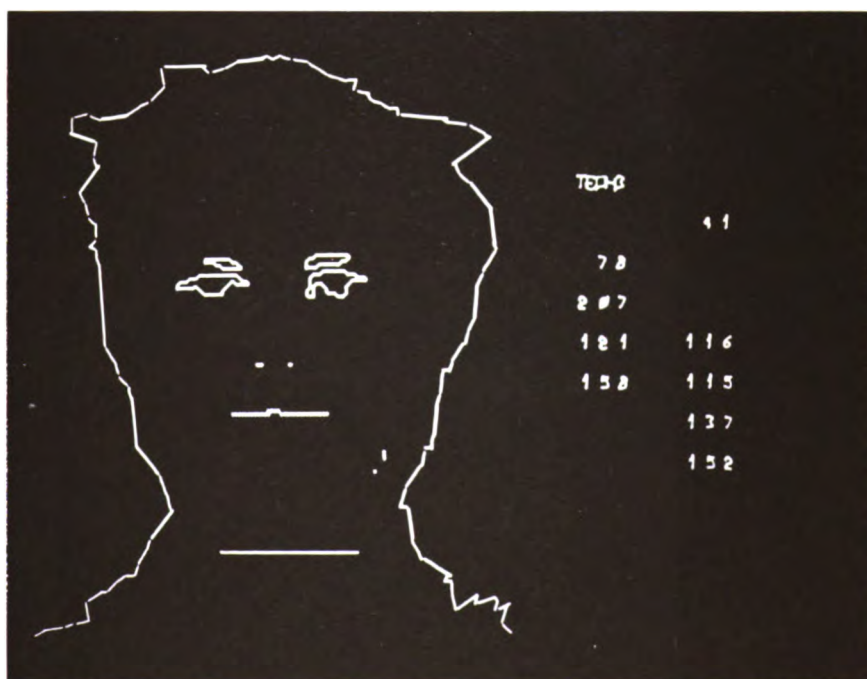


(b)



(a)

Figure 10-28

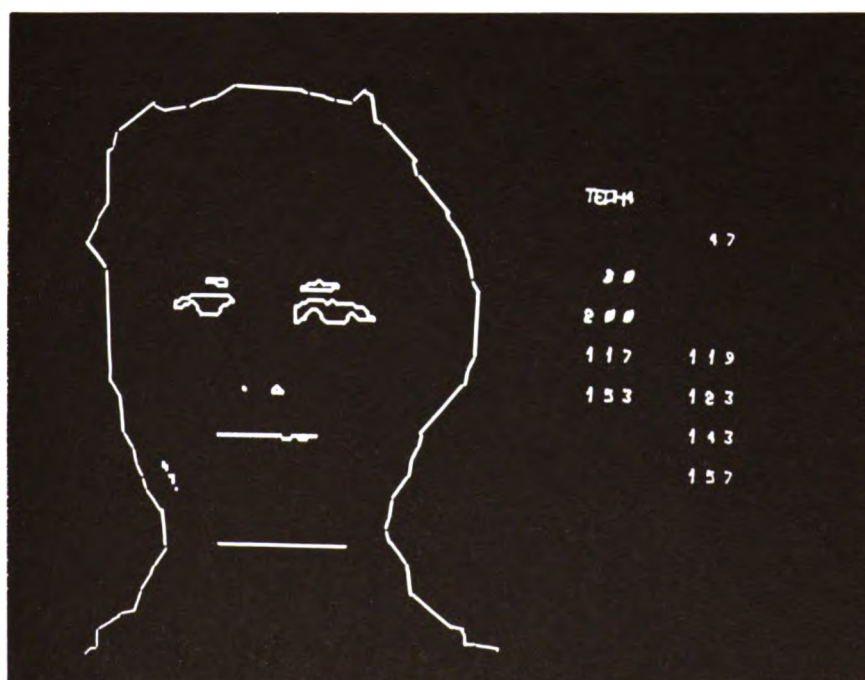


(b)



(a)

Figure 10-29



(b)

which features,

The test of identification of people was performed as follows. First, the complete collection of pictures was processed, and all of the measurements used for recognition were recorded. This produced 24 sets of measurements describing the 10 people. These measurements are given in Table 10-2. Recognition was attempted on these sets of measurements.

All of the measurements were normalized to compensate for the varying distances of the subjects from the camera as described earlier. This normalization was performed by dividing all of the measurements in each picture by the height measured in that picture. Normalized measurements are given in Table 10-3.

Weights for use in the classification procedure were obtained as described in chapter 9. These weights are listed in Table 10-4.

For each set of measurements, the remainder of the collection of measurements was considered to be a dictionary of known individuals. Thus, the following experiment was repeated 24 times. An unknown measurement vector was selected, and an attempt was made to identify the person to whom these measurements belonged from the dictionary which contained 23 entries.

In this recognition experiment only two errors were made. The correct individual was identified in 91% of the

TABLE 10-2,
Measurements for identification test.

Each entry consists of a 3 letter code for the individual followed by an identifying letter or digit. The ten measurements which follow are the ten measurements listed in Figures 2-1 and 2-2.

ELM1					
185.	21.0	14.0	52.0	46.0	134.
44.0	88.0	25.0	45.0		
ELM2					
185.	20.0	14.0	51.0	38.0	133.
42.0	89.0	23.0	42.0		
GJG1					
184.	21.0	16.0	59.1	41.0	137.
43.0	88.0	25.0	44.0		
GJG2					
182.	24.0	15.0	41.0	40.0	135.
41.0	88.0	24.0	42.0		
JLR1					
195.	23.0	14.0	58.0	42.0	145.
52.2	108.	28.0	44.0		
JLR2					
194.	23.0	16.0	57.0	42.0	138.
51.1	108.	29.0	46.0		
JZC1					
189.	23.0	15.0	52.0	41.0	145.
55.0	91.0	23.0	42.0		
JZC2					
190.	23.0	15.0	55.0	40.0	151.
53.0	88.0	21.0	40.0		
KEL1					
168.	19.0	11.0	44.0	36.0	115.
40.0	75.0	21.0	38.0		
KEL4					
173.	21.0	11.0	63.0	44.0	119.
40.0	69.0	19.0	35.0		
KEL5					
172.	20.0	11.0	47.0	43.0	120.
42.0	69.0	20.0	36.0		
LDE1					
154.	17.0	10.0	47.0	50.0	113.
39.0	68.0	24.0	39.0		
LDE2					
156.	17.0	11.0	47.0	45.0	114.
40.0	76.0	24.0	40.0		

TABLE 10-2, (Continued)
Measurements for Identification test,

RAJ1					
174.	18.0	13.0	50.0	27.0	114.
38.3	68.0	22.0	26.0		
RAJ2					
177.	18.0	12.0	50.0	28.0	120.
39.2	79.0	19.0	25.0		
RGG7					
183.	22.0	16.0	45.0	39.0	129.
46.0	80.0	18.0	36.0		
RGG8					
190.	21.0	16.0	58.0	40.0	135.
44.0	86.0	20.0	40.0		
RGGC					
190.	21.0	15.0	55.0	40.0	132.
48.0	82.0	18.0	41.0		
RPH2					
174.	20.0	13.0	46.0	47.0	116.
39.1	73.0	14.0	30.0		
RPH3					
172.	19.0	14.0	52.0	48.0	112.
41.0	71.0	15.0	30.0		
TED1					
162.	20.0	13.0	42.0	34.0	117.
35.0	82.0	24.0	37.0		
TED2					
163.	20.0	13.0	43.1	33.1	120.
37.1	89.0	24.0	38.0		
TED3					
163.	19.0	13.0	39.0	50.0	129.
37.0	74.0	22.0	37.0		
TED5					
154.	18.0	12.0	39.0	31.0	120.
36.2	74.0	22.0	36.0		

TABLE 10-3,
Normalized measurements,

The format is the same as Table 10-2.

ELM1					
1.00	0.114	0.757E-01	0.281	0.249	0.724
0.238	0.476	0.135	0.243		
ELM2					
1.00	0.108	0.757E-01	0.276	0.205	0.719
0.227	0.481	0.124	0.227		
GJG1					
1.00	0.114	0.870E-01	0.321	0.223	0.745
0.234	0.478	0.136	0.239		
GJG2					
1.00	0.132	0.824E-01	0.225	0.220	0.742
0.225	0.484	0.132	0.231		
JLR1					
1.00	0.118	0.718E-01	0.297	0.215	0.744
0.268	0.554	0.144	0.226		
JLR2					
1.00	0.119	0.825E-01	0.294	0.216	0.711
0.263	0.557	0.149	0.237		
JZC1					
1.00	0.122	0.794E-01	0.275	0.217	0.767
0.291	0.481	0.122	0.222		
JZC2					
1.00	0.121	0.789E-01	0.289	0.211	0.795
0.279	0.463	0.111	0.211		
KEL1					
1.00	0.113	0.655E-01	0.262	0.214	0.685
0.238	0.446	0.125	0.226		
KEL4					
1.00	0.121	0.636E-01	0.364	0.254	0.688
0.231	0.399	0.110	0.202		
KEL5					
1.00	0.116	0.640E-01	0.273	0.250	0.698
0.244	0.401	0.116	0.209		
LDE1					
1.00	0.110	0.649E-01	0.305	0.325	0.734
0.253	0.442	0.156	0.253		
LDE2					
1.00	0.109	0.705E-01	0.301	0.288	0.731
0.256	0.487	0.154	0.256		

TABLE 10-3. (Continued)

Normalized measurements,

RAJ1					
1.00	0.103	0.747E-01	0.287	0.155	0.655
0.220	0.391	0.126	0.149		
RAJ2					
1.00	0.102	0.678E-01	0.282	0.158	0.678
0.221	0.446	0.107	0.141		
RGG7					
1.00	0.120	0.874E-01	0.246	0.213	0.705
0.251	0.437	0.984E-01	0.197		
RGG8					
1.00	0.111	0.842E-01	0.305	0.211	0.711
0.232	0.453	0.105	0.211		
RGGC					
1.00	0.111	0.789E-01	0.289	0.211	0.695
0.253	0.432	0.947E-01	0.216		
RPH2					
1.00	0.115	0.747E-01	0.264	0.270	0.667
0.225	0.420	0.805E-01	0.172		
RPH3					
1.00	0.110	0.814E-01	0.302	0.279	0.651
0.238	0.413	0.872E-01	0.174		
TED1					
1.00	0.123	0.802E-01	0.259	0.210	0.722
0.216	0.506	0.148	0.228		
TED2					
1.00	0.123	0.798E-01	0.264	0.203	0.736
0.228	0.546	0.147	0.233		
TED3					
1.00	0.117	0.798E-01	0.239	0.307	0.791
0.227	0.454	0.135	0.227		
TED5					
1.00	0.117	0.779E-01	0.253	0.201	0.779
0.235	0.481	0.143	0.234		

TABLE 10-4, Weights.

<u>Measurement</u>	<u>σ^2</u>	<u>σ</u>
From body picture:		
Head width	0.0000402	0.0063405
Neck width	0.0000197	0.0044354
Shoulder width	0.0019402	0.0440476
Hip width	0.0019666	0.0443458
From head picture:		
Head width	0.0007326	0.0270665
Distance between eyes	0.0001170	0.0108170
Top of head to eyes	0.0014393	0.0379384
Eyes to nose	0.0000663	0.0081423
Eyes to mouth	0.0001208	0.0109927

cases involved. The distance in the recognition space between each pair of measurements was tabulated and is given in Table 10-5. This test indicates that the identification of people is possible relying solely on measurements obtained by a computer.

The two errors in the recognition tests occurred when one individual was misidentified twice. The reason for this mistake was faulty measurements of the width of the shoulders and neck in the picture of the body. The faulty measurements were caused by low contrast with the background.

An example of this sort of problem can be seen in Figure 10-1(a) where there is very little contrast between the subjects shirt and the window behind him.

FUTURE WORK.

The recognition procedures described in this thesis are elaborate and complex. It would be worthwhile to examine these procedures in detail and attempt to determine the key heuristics on which success or failure of the algorithms really depends. In isolation they could be examined more effectively to determine the range of their usefulness and applicability. Statistical data could be obtained on the success rates of particular heuristics. This could then be used to obtain meaningful evaluations. Unfortunately, such testing and evaluation is a problem which is comparable. In

TABLE 10-5. RECOGNITION DISTANCES.

ELM1	ELM2	GJG1	GJG2	JLR1	JLR2	JZC1	JZC2	KEL1	KEL4	KEL5	LDE1
ELM1	0.0	6.7	8.5	16.0	18.0	36.0	42.0	12.8	42.4	27.3	19.6
ELM2	6.7	0.0	13.1	19.5	27.7	31.0	43.8	9.5	31.5	20.8	41.6
GJG1	8.5	13.1	0.0	15.1	28.6	18.1	39.6	34.3	60.8	49.8	42.4
GJG2	16.0	19.5	15.1	0.0	34.4	29.3	44.3	31.8	54.6	42.9	56.7
JLR1	18.0	27.7	28.6	34.4	0.0	9.0	19.9	28.7	60.8	41.4	29.2
JLR2	31.0	31.0	39.6	0.0	9.0	0.0	29.1	40.9	82.3	61.6	36.8
JZC1	43.8	43.8	44.3	19.9	29.1	0.0	5.7	46.1	66.7	45.0	60.5
JZC2	41.1	42.6	44.7	31.6	47.2	5.7	0.0	47.4	54.3	39.3	76.7
KEL1	9.5	34.3	31.8	28.7	40.9	46.1	47.4	0.0	18.3	6.6	33.1
KEL4	31.5	60.8	54.6	60.8	82.3	66.7	54.3	18.3	0.0	7.5	69.1
KEL5	20.8	49.8	42.9	41.4	61.6	45.0	39.3	6.6	0.0	0.0	47.5
LDE1	19.6	41.6	42.4	56.7	36.8	60.5	76.7	33.1	69.1	47.5	0.0
LDE2	12.7	33.1	28.9	46.8	19.9	48.3	66.5	32.4	75.7	52.2	4.0
RAJ1	95.3	63.5	99.1	98.9	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	28.2	30.9	28.2	84.0
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	26.6	28.4	21.8	86.8
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	30.5	60.2	49.3	44.9
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0	27.0	32.0	67.1	53.0	41.0
RPH2	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RPH3	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RAJ1	95.3	63.5	98.9	108.6	121.8	121.4	104.7	63.7	52.2	53.4	142.7
RAJ2	114.4	74.0	125.0	119.2	147.7	129.0	104.0	72.2	52.2	56.3	168.9
RGG1	50.8	35.6	46.0	40.6	66.7	37.5	26.4	45.9	44.1	37.0	113.7
RGG2	28.2	12.9	31.2	53.0	53.4	45.1	33.6	27.0	30.5	60.2	49.3
RGG3	36.8	23.2	42.2	48.1	63.5	36.0					

the effort required, to the original design and development of the system.

The work reported in this thesis could be extended to deal with larger sets of people. One way in which this could be done would be to use absolute measurements of people rather than measurements which have been normalized by height. Absolute measurements can be obtained from the pictures in a straight-forward way if the distance from camera to subject is known. Alternatively, if the geometry of the camera position is known, the distance from the camera to the individual may be calculated using the ground plane assumption (Sobel [1970]). In the pictures that were processed, the height of the people could be correctly measured to within 2 raster units. This corresponds to about a half of an inch when considered as a fraction of a person's height. Thus, a preliminary reduction in the fraction of the dictionary to be searched could be done using the absolute height of the person. The rest of the measurements obtained from the pictures could be used to distinguish people of almost the same height. In this way the number of people that a recognition system could handle could be greatly expanded.

Another valuable extension of this work would be to attempt to develop a formal structure that would permit high-level specification of the model and the heuristics which are buried in the program that has been described.

Possible lines of approach could be found in the attempts at linguistic description of pictures and formalization of models which were discussed in Chapter 4. Such a high-level specification, if successful in reproducing the effect of the algorithms so far described, could be applied to other areas of picture processing to test its generality.

It would be useful to attempt to overcome the obstacles that prevented this program from performing on-line recognition. What is needed is a picture input system with the ability to digitize a wider range of light intensities. Also needed in an effective on-line system is a quick-response picture output device for digital pictures that can provide visual feedback to the operator while setting input parameters.

It is clear that this heuristic program is nowhere near the final solution to the problem of visual identification of people. Nevertheless, the success that has been achieved indicates that the methods used are worthwhile. The research which has been done here provides useful procedures for further work in processing very complex scenes.

References

- Abbamonte, M., Johnston, E.G., Lee, Y.H., Nagel, R., Rosenfeld, A., and Thurston, M. [1970] Edge and curve enhancement in digital pictures, 2. Report 70-103, Computer Science Center, University of Maryland, College Park, Maryland (January).
- Bell, D.A. [1968] Computer aided design of image processing techniques. Proc. of I.E.E./N.P.L. Conference on Pattern Recognition, Institute of Electrical Engineers, London.
- Billingsley, F.C. [1967] Digital image processing at JPL. Proc. Computer Imaging Techniques Seminar, Society of Photo-optical Instrumentation Engineers, Washington, D.C.
- Bisson, C.L. [1965a] Preliminary investigation on measurements by computer of the distances on and about the eyes. Report PRI:18, Panoramic Research Inc., Palo Alto, California (April).
- Bisson, C.L. [1965b] Location of some facial features by computer. Report PRI:20, Panoramic Research Inc., Palo Alto, California (June).
- Bledsoe, W.W. [1964] The model method in facial recognition. Report PRI:15, Panoramic Research Inc., Palo Alto, California (August).
- Bledsoe, W.W. [1966] Man-machine facial recognition. Report PRI:22, Panoramic Research Inc., Palo Alto, California (August).
- British Computer Society. [1967] Character Recognition 1967. British Computer Society, London.
- Clowes, M.B. [1969] Pictorial relationships - a syntactic approach. In Machine Intelligence 4, Meltzer, B. and Michie, D. (Ed.), American Elsevier, New York, pp. 361-383.
- Cover, T.M. and Hart, P.E. [1967] Nearest neighbor pattern classification. IEEE Transactions on Information Theory IT-13 (January), pp. 21-27.
- El'bur, R.E. [1967] Utilization of the apparatus of projective geometry in the process of the identification of individuals by their photographs. In Problems of

Cybernetics and Law, Kudryavtsev, V.N. (Ed.), Nauka Publishing House, Moscow, pp. 321-348. Available through U.S. Department of Commerce, Joint Publications Research Service, No. JPRS: 43,954 (January 10, 1968).

Falk, G. [1969] Some implications of planarity for machine perception. Artificial Intelligence Memo No. 107, Stanford University, Stanford, California (December).

Felgenbaum, E.A. and Feldman, J. (Ed.) [1963] Computers and Thought, McGraw-Hill, New York.

Feldman, J.A., Feldman, G.M., Falk, G., Grape, G., Pearlman, J., Sobel, I., Tenenbaum, J.M. [1969] The Stanford hand-eye project. Proc. International Joint Conference on Artificial Intelligence (May), pp. 521-526.

Feldman, J.A. and Gries, D. [1968] Translator writing systems. Communications of the ACM 11 (February), pp. 77-113.

Forsen, G.E. [1968] Processing visual data with an automaton eye. In Pictorial Pattern Recognition, Cheng, G.C. et al (Ed.), Thompson Book Co., Washington, D.C.

Greenblatt, R.D., Eastlake, D.E., Crocker, S.D. [1967] The Greenblatt chess program. Proc. AFIPS 1967 Fall Joint Computer Conference, Thompson Book Co., Washington, D.C., pp. 801-810.

Greenblatt, R.D., Holloway, J., Sordillo, D.A. [1966] Sides 21. Artif. Intell. Memo No. 104-2 (MAC-M-320), M.I.T., Cambridge, Mass. (Aug).

Guzman, A. [1968] Decomposition of a visual scene into three-dimensional bodies. Proc. AFIPS 1968 Fall Joint Computer Conference, Vol. 33, Thompson Book Co., Washington, D.C., pp. 291-304.

Hart, P.E. [1969] Three object recognition methods using projective invariants. Report, Stanford Research Institute, Menlo Park, California (January).

Hueckel, M. [1969] An operator which locates edges in digitized pictures. Artificial Intelligence Memo No. 105, Stanford Univ., Stanford, Calif. (October).

Ho, Y. and Agrawala, A.K. [1968] On pattern classification algorithms, introduction and survey. Proc. IEEE 56 (December), pp. 2101-2114.

Kelly, M.D. [1970] Edge detection in pictures by computer

- using planning. Artificial Intelligence Memo No. 108, Stanford University, Stanford, California (January).
- Kirsch, R.A., Cahn, L., Ray, C., and Urban, G.H. [1957] Experiments in processing pictorial information with a digital computer. Proc. 1957 Eastern Joint Computer Conference, Spartan Books, New York, pp. 221-229.
- Lederberg, J. and Feigenbaum, E.A. [1968] Mechanization of inductive inference in organic chemistry. In Formal Representation for Human Judgement, Kleimuntz, B. (Ed.), Wiley, New York.
- Ledley, R.S., Rotolo, L.S., Golab, T.J., Jacobsen, J.D., Ginsberg, M.D., and Wilson, J.B. [1965] FIDAC: film input to digital automatic computer and associated syntax-directed pattern recognition programming system. In Optical and Electro-Optical Information Processing, Tippet, J.T. et al (Ed.), MIT Press, Cambridge, Mass.
- Lipkin, L.E., Watt, W.C., and Kirsch, R.A. [1966] The analysis, synthesis, and description of biological images. Annals of the New York Academy of Sciences 128 (January), pp. 984-1012.
- Lourie, J.R. [1969] The computation of connected regions in interactive graphics. Proc. 1969 A.C.M. National Conference, Association for Computing Machinery, New York, pp. 369-377.
- McCarthy, J., Earnest, L.D., Reddy, D.R., and Vicens, P.J. [1968] A computer with hands, eyes, and ears. Proc. AFIPS 1968 Fall Joint Computer Conference, Vol. 33, Thompson Book Co., Washington, D.C., pp. 329-338.
- Miller, W.F. and Shaw, A.C. [1967] A picture calculus, GSG Memo 40, Computation Group, Stanford Linear Accelerator Center, Stanford, California (June).
- Miller, W.F. and Shaw, A.C. [1968] Linguistic methods in picture processing: a survey. Proc. AFIPS 1968 Fall Joint Computer Conference, Vol. 33, Thompson Book Co., Washington, D.C., pp. 279-290.
- Minsky, M. [1961] Steps toward artificial intelligence. Proc. of the IRE 49 (January), pp. 8-30. Also in Computers and Thought, Feigenbaum, E.A. and Feldman, J. (Ed.), McGraw-Hill, New York, 1963, pp. 406-450.
- Minsky, M. and Papert, S. [1969] Perceptrons. MIT Press, Cambridge, Massachusetts.

- Moorer, A. [1969] Stanford A-I Project monitor manual, Artificial Intelligence Operating Notes Nos. 54 and 55, Stanford University, Stanford, California (September).
- Nagy, G. [1968] State of the art in pattern recognition. Proc. IEEE 56 (May), pp. 836-862.
- Narasimhan, R. [1966] Syntax-directed interpretation of classes of pictures. Communications of the ACM 9 (March), pp. 166-173.
- Narasimhan, R. and Forango, J.P. [1964] Some further experiments in the parallel processing of pictures. IEEE Transactions on Electronic Computers 13 (December), pp. 748-750.
- Newell, A., Shaw, J.C., and Simon, H.A. [1959] Report on a general-problem solving program. Proc. International Conference on Information Processing, UNESCO House, Paris, pp. 256-264.
- Nilsson, N.J. [1969] A mobile automaton: an application of artificial intelligence techniques. Proc. International Joint Conference on Artificial Intelligence (May), pp. 509-515.
- Paul, R., Falk, G., and Feldman, J.A. [1969] The computer representation of simply described scenes. Artificial Intelligence Memo No. 101, Stanford Univ., Stanford, Calif. (October).
- Pfaltz, J.L. and Rosenfeld, A. [1969] Web grammars. TR-69-84, Computer Science Center, University of Maryland, College Park, Maryland (January).
- Pingale, K.K. [1966] A program to find objects in a picture. Artificial Intelligence Memo No. 39, Stanford Univ., Stanford, Calif. (January).
- Pingale, K.K. [1969] Visual perception by a computer. In Automatic Interpretation and Classification of Images, Grasselli, A. (Ed.), Academic Press, New York, pp. 277-284.
- Pingale, K.K., Singer, J.A., Wichman, W.M. [1968] Computer control of a mechanical arm through visual input. Proc. IFIP Conference, Edinburgh, pp. H140-H146.
- Rabinow, J. [1968] The present state of the art in reading machines. In Pattern Recognition, Kanal, L.N. (Ed.), Thompson Book Co., Washington, D.C., pp 3-29.

- Raphael, B. [1968] Programming a robot. Proc. IFIP Conference, Edinburgh, pp. H135-H139.
- Rastrigin, L.A. [1967] About the identification of images of spatial objects. In Problems of Cybernetics and Law, Kudryavtsev, V.N. (Ed.), Nauka Publishing House, Moscow, pp. 361-368. Available through U.S. Department of Commerce, Joint Publications Research Service, No. JPRS: 43,954 (January 10, 1968).
- Reddy, D.R. [1969] On the use of environmental, syntactic, and probabilistic constraints in vision and speech. Artificial Intelligence Memo No. 78, Stanford University, Stanford, California (January).
- Roberts, L.G. [1963] Machine perception of three-dimensional solids. Technical Report No. 315, Lincoln Laboratory, M.I.T. (May). Also in Optical and Electro-Optical Information Processing, Tippett, J.T. et al (Ed.), MIT Press, Cambridge, Mass., 1965, pp. 159-197.
- Rosenfeld, A. [1969a] Picture Processing by Computer. Academic Press, New York.
- Rosenfeld, A., Thomas, R.B., and Lee, Y.H. [1969b] Edge and curve enhancement in digital pictures. Univ. of Maryland Computer Science Center Technical Report 69-93 (May).
- Rosenfeld, A. [1970] Connectivity in digital pictures. Journal of the ACM 17 (January), pp. 146-160.
- Sakai, T., Nagao, M., and Fujibayashi, S. [1969] Line extraction and pattern detection in a photograph. Pattern Recognition 1 (March), pp. 233-248.
- Samuel, A.L. [1967] Some studies in machine learning using the game of checkers II - recent progress. IBM Journal of Research and Development 11 (November), pp. 601-617.
- Shaw, A.C. [1968] The formal description and parsing of pictures. PhD Thesis, Report CS 94, Computer Science Department, Stanford Univ., Stanford, Calif.
- Sobel, I. [1970] Camera models and machine perception. PhD Thesis, Artificial Intelligence Memo No. 121, Stanford University, Stanford, California (May).
- Tenenbaum, J.M. [1970] Accomodation in computer vision. Forthcoming Ph.D. Thesis, Electrical Engineering Dept., Stanford University.

Thorwald, J. [1965] The Century of the Detective, Harcourt, Brace, & World, New York.

Williams, C.M. [1965] Isolation of important features of a multi-toned picture. Artificial Intelligence memo No. 29, Stanford University, Stanford, California (January).

Yurans, V. [1967] Certain questions concerning the theory of identification of objects with the use of the apparatus of projective geometry. In Problems of Cybernetics and Law, Kudryavtsev, V.N. (Ed.), Nauka Publishing House, Moscow, pp. 349-360. Available through U.S. Department of Commerce, Joint Publications Research Service, No. JPRS: 43,954 (January 10, 1968).

APPENDIX A

Two pictures of bodies: Input data

[illegible]

[illegible]

Digitized by Google

APPENDIX B

Two pictures of heads: Input data

181

[illegible]

[illegible]

[illegible]

Digitized by Google

Digitized by Google

193

COEEEEEEDC8B8CCCCCDDC9665599977778.....
BCDDDEEDDCCCB8B9ACDDC8555799986998.....
B8CDDDDDDCCCB8B9ACDDC8555799986998.....
9AB8CDDDDCCCB8B9ACDDC8555799986998.....
98B8CDDDDCCCB8B9ACDDC8555799986998.....
13299AAAB8B8C8B9ACDDC8555799986998.....
89899AAAB8B8C8B9ACDDC8555799986998.....
89899AAAB8B8C8B9ACDDC8555799986998.....
9678899AAAB8B8C8B9ACDDC8555799986998.....
6677888999AAAB8B8C8B9ACDDC8555799986998.....
7777888999AAAB8B8C8B9ACDDC8555799986998.....
866777888999AAAB8B8C8B9ACDDC8555799986998.....
98766677888999AAAB8B8C8B9ACDDC8555799986998.....
A876666677888999AAAB8B8C8B9ACDDC8555799986998.....
A976666677888999AAAB8B8C8B9ACDDC8555799986998.....
142A96666677888999AAAB8B8C8B9ACDDC8555799986998.....
9977666677888999AAAB8B8C8B9ACDDC8555799986998.....
9A87666777888999AAAB8B8C8B9ACDDC8555799986998.....
889976777888999AAAB8B8C8B9ACDDC8555799986998.....
897876777888999AAAB8B8C8B9ACDDC8555799986998.....
7667799667888999AAAB8B8C8B9ACDDC8555799986998.....
655668888999AAAB8B8C8B9ACDDC8555799986998.....
4456788999AAAB8B8C8B9ACDDC8555799986998.....
12455677888999AAAB8B8C8B9ACDDC8555799986998.....
1134566677888999AAAB8B8C8B9ACDDC8555799986998.....
152223556677888999AAAB8B8C8B9ACDDC8555799986998.....
544556777888999AAAB8B8C8B9ACDDC8555799986998.....
654556777888999AAAB8B8C8B9ACDDC8555799986998.....
7655566777888999AAAB8B8C8B9ACDDC8555799986998.....
6555566777888999AAAB8B8C8B9ACDDC8555799986998.....
7665556667788999AAAB8B8C8B9ACDDC8555799986998.....
6665556667788999AAAB8B8C8B9ACDDC8555799986998.....
66555566677999AAAB8B8C8B9ACDDC8555799986998.....
66555566677999AAAB8B8C8B9ACDDC8555799986998.....
66555566677999AAAB8B8C8B9ACDDC8555799986998.....
1606655556667789AC.....
66655555566789AC.....
6665555556789AC.....
6666655556789AC.....
66666655568999AA.....
666666666699AA9A.....
6656666666899999D.....
6656666677899999C.....
666666777999999D.....
666666778889999E.....
567666788889999F.....
17567666788899999E.....
56676899999999E.....
56678999999999E.....
56677876969999AD.....
56676769789999AD.....
56667676677889A.....
56676666667889A.....
55666566666889A.....
55665556656889A.....


```

555555555555556789A.....
18055555555555556789A.....
555555555555556689A.....
55455455555555679A.....
4435544555555679A.....
53455555555555689E.....
334445544555689C.....
3344445555556789B.....
3343345555556789A.....
3332345555556688A.....
3332334455556678A.....
1903333333344456668A.....
33323234555555658.....
333223555445449.....
322233334443AB.....
222232334333B.....
222223223325C.....
2222222222249.....
1111222222280.....
11112111115A.....
1111111111A.....
200111111111240.....
1111111128.....
11111111AD.....
01111115AA.....
11110129.....
1111128E.....
111119C.....
11015R.....
1115B.....
1129E.....
21011768A.....
1A86459E.....
2A8653A.....
AC654324BR.....
BC755123B8D.....
DC86542318.....
CF67653222R.....
CB766432225.....
BR766542224.....
R9665542224.....
2297555432222.....
86555442222D.....
8655543334D.....
86566554334D.....
76666554444D.....
6666655445E.....
66776605556.....
66889867668.....
78999886669.....
799AAA9889.....
2309088BBAAAB.....

```


APPENDIX C

Programs for locating the outline of the head


```

00100 SUBROUTINE HAIN
00200 MAIN CONTROL FOR HEAD PROCESSING.
00300 IMPLICIT INTEGER (A-Z)
00400 COMMON /HEDMES/ XL,XR,YT,MEASX(4),MEASY(4)
00500 COMMON /CON/ ADUM(3),BITS,IWID,LINLEN,FLINE,LLINE,LSIDE,RSIDE,
00600 1 TVPNTR,TVSIZ,NPERWD,ILEN,BDUM(8),ITEXT(10)
00700 INTEGER BITS,FLINE,RSIDE,IVPNTR,TVSIZ
00800 COMMON /PTRS/ KPTR,NPTR,IPTR(20)
00900 COMMON /FLAGS/ NUTN1,PICNA1,INVERT,NUTN2,KEEPLP,NUTN3(3),
01000 1 USETV,KG1G2,NUTN4(40)
01100 COMMON /OUTFGS/ IFDITY,IFDLPT,IFDDPY,IYORN
01200 DATA OUTLINEONLY/.FALSE./
01300
01400 CALL GIVEUP(-1)
01500 CALL KILLLL
01600
01700 HEDPOG=IADPST(400)
01800
01900 READ IN PICTURE OF HEAD.
02000 IF (USETV.NE.0) GO TO 100
02100 IF (PICNA1.NE.0) PICNA1=3
02200 JP=INDAT(0)
02300 GO TO 200
02400 CALL TVSET(LENPB)
02500 CALL GIVEUP(-1)
02600 JP=INTV(0)
02700 CALL DECDMX(IPTR(JP))
02800 JP IS ORIGINAL HEAD.
02900
03000 INITIALIZE G1 AND G2.
03100 CALL FILCON(JP,ADUM)
03200 KG1G2=-1
03300 CALL G1(LSIDE)
03400 CALL G2(FLINE)
03500 KG1G2=0
03600
03700 JP2=IREO3(JP,8)
03800 JP2 IS JP REDUCED IN SIZE BY A FACTOR OF 8.
03900 KEEPPL=-1
04000 IF (KRAPPE.NE.0) CALL PRTPIC(0)
04100
04200 JP3=JSURE3(JP2)
04300 JP3 IS ALL OF THE EDGES FROM JP2.
04400 IF (IFDDPY.EQ.0) GO TO 330
04500 PLNPOG=IADPST(1000)
04600 CALL OSLINS(JP3,250,450,PLNPOG)
04700 INVERT=-1
04800 IF (KRAPPE.NE.0) CALL PRTPIC(JP3)
04900
05000 JP4=IHEDED(JP3)
05100 JP4 IS THE EDGES FROM JP3 WHICH FORM THE BORDER OF THE

```



```

17-JUL-70 0738 HAIN4 1,MDK
05100 C HEAD.
05200 IF (IFDDPY,EO,0) GO TO 430
05300 CALL SETPOG(PLNPOG)
05400 CALL DSLINS(JP4,250,250,PLNPOG)
05500 INVERT=-1
05600 IF (KRAPPE,NE,0) CALL PRTPIC(JP4)
05700
05800 JP5=JBIGHE(JP,HEDPOG)
05900 JP5 CONTAINS THE EDGES IN THE ORIGINAL PICTURE WHICH
06000 FORM THE BORDER OF THE HEAD.
06100 IF (IFDDPY,NE,0) CALL DPOUT(HEDPOG)
06200 CALL PRTPIC(JP)
06300 CALL PRTPIC(JP5)
06400 INVERT=0
06500
06600 C IDENTIFY BOUNDARIES OF THE HEAD.
06700 CALL ISIL(XL,XR,YT,DUMMY)
06800
06900 C RELEASE BUFFERS.
07000 CALL IGIVEU(JP5)
07100 CALL IGIVEU(JP4)
07200 IF (IFDDPY,EO,0) GO TO 730
07300 CALL HYDPOG(PLNPOG)
07400 CALL IGIVEX(PLNPOG)
07500 CALL IGIVEU(JP3)
07600 CALL IGIVEU(JP2)
07640
07680 IF (OUTLINEONLY) RETURN
07720
07800 C FIND FACIAL FEATURES.
07900 KPTR=JP
08000 CALL FDFEAT(HEDPOG)
08100
08200 END

```



```

20120 FUNCTION JSURE3(ISPTR)
20220 IMPLICIT INTEGER (A-Z)
20320 COMMON /CON/ ADUM(3),BITS,IWID,LINLEN,FLINE,LLINE,LSIDE,RSIDE,
20420 1 TVPNTR,TVSIZ,NPERWD,ILEN,BDUM(8),ITEXT(10)
20520 INTEGER BITS,FLINE,RSIDE,TVPNTR,TVSIZ
20620
20720 IF (SWITCH) IDPTR=INITOP(ISPTR,3,'J SURE 3')
20820 IF (.NOT.SWITCH) IDPTR=INITOP(ISPTR,2,'2X2 MAR5')
20920
21020 T=1
21120
21220 DO 500 J=FLINE,LLINE
21320 DO 500 I=LSIDE,RSIDE
21420 IF (.NOT.SWITCH) GO TO 100
21520 IVAL=JEDGE3(I,J,-1)
21620 GO TO 460
21720
21820 1 1 2 - 4 / 8 \
21920 C
22020 A B
22120 C D
22220
22320 A=LOAD(I,J)
22420 R=LOAD(I+1,J)
22520 C=LOAD(I,J+1)
22620 D=LOAD(I+1,J+1)
22720 IF (A-B) 140,220,120
22820 A,GT,B
22920 IF (MIN0(A,C)-MAX0(B,D),GT,T) GO TO 401
23020 GO TO 200
23120 IF (MIN0(B,D)-MAX0(A,C),GT,T) GO TO 401
23220 IF (A-C) 240,300,220
23320 A,GT,C
23420 IF (MIN0(A,B)-MAX0(C,D),GT,T) GO TO 402
23520 GO TO 300
23620 IF (MIN0(C,D)-MAX0(A,B),GT,T) GO TO 402
23720 CONTINUE
23820 A:D
23920 IF (IABS(A-D),LE,T) GO TO 350
24020 IF (A-D) 330,350,310
24120 C
24220 A,GT,D
24320 IF (A-MAX0(B,C,D),GT,T) GO TO 404
24420 IF (MIN0(A,B,C)-D,GT,T) GO TO 404
24520 GO TO 350
24620 C
24720 A,LT,D
24820 IF (D-MAX0(A,B,C),GT,T) GO TO 404
24920 IF (MIN0(B,C,D)-A,GT,T) GO TO 404
25020 B:C
25120 IF (IABS(B-C),LE,T) GO TO 450
25220 IF (B-C) 380,450,360
25320 C
25420 B,GT,C
25520 IF (B-MAX0(A,C,D),GT,T) GO TO 408

```


17-JUL-70	0833	BOR34	1,MDK
25100		IF (MIN0(A,B,D)-C,GT,T) GO TO 408	
25200		GO TO 450	
25300	C	B,LT,C	
25400	380	IF (C-MAX0(A,B,D),GT,T) GO TO 408	
25500		IF (MIN0(A,C,D)-B,GT,T) GO TO 408	
25600		GO TO 450	
25700	401	IVAL=1	
25800		GO TO 460	
25900	402	IVAL=2	
26000		GO TO 460	
26100	404	IVAL=4	
26200		GO TO 460	
26300	408	IVAL=8	
26400		GO TO 460	
26500			
26600	450	IVAL=0	
26700	460	CALL STORE(I,J,IVAL)	
26800	500	CONTINUE	
26900			
27000		JSURE3=IDPTR	
27100		END	


```

00100      FUNCTION INITOP(SPTR,N,NAME)
00200      INITIALIZE FOR NON OPERATOR.
00300      IMPLICIT INTEGER (A-Z)
00400      DIMENSION NAME(2)
00500      COMMON /CON/ A(3),BITS,IWID,LINLEN,FLINE,LLINE,LSIDE,RSIDE,
00600      1      TVPNTR,TVSIZE,NPERWD,ILEN,B(8),ITEXT(10)
00700      DIMENSION Z(32)
00800
00900      CALL FILCON(SPTR,A(1))
01000      CALL INLOAD(SPTR)
01100      DO 100 I=1,32
01200      Z(I)=A(I)
01300      ALLOCATE 1 WORD PICTURE BUFFER.
01400      DPTR=IALLOC(1)
01500      CALL FILCON(DPTR,A(1))
01600      BITS=Z(4)
01700      IWID=Z(5)-(N-1)
01800      NPERWD=Z(13)
01900      LINLEN=(IWID+NPERWD-1)/NPERWD
02000      FLINF=Z(7)+(N-1)/2
02100      LLINE=Z(8)-N/2
02200      LSIDE=Z(9)+(N-1)/2
02300      RSIDE=Z(10)-N/2
02400      ILEN=Z(14)-(N-1)
02500      ITEXT(1)=Z(23)
02600      ITEXT(2)=Z(24)
02700      ITEXT(3)=NAME(1)
02800      ITEXT(4)=NAME(2)
02900      CALL PUTCON(DPTR,A(1))
03000      NOW ALLOCATE NEEDED LENGTH.
03100      CALL ALOCAD(DPTR,LINLEN*ILEN)
03200      CALL INSTOR(DPTR)
03300      CALL FILCON(DPTR,A(1))
03400      INITOP=DPTR
03500      END

```



```

00100 FUNCTION IHEDED(ISPTR)
00200 IMPLICIT INTEGER(A-Z)
00300 COMMON /CON/ A(3),BITS,IWID,LINLEN,FLINE,LLINE,LSIDE,RSIDE,
00400 1 TVPNT,TVSIZ,NPERWD,ILEN,B(8),ITEXT(10)
00500 COMMON /SENTS/ VBL,VLSB,VLST,VTL,VTR,VRST,VRSB,VBR
00600
00700 LIST STRUCTURE ENTRIES:
00800
00900 VTL VTR
01000
01100 VLST VRST
01200
01300 VLSB VRSB
01400
01500 VBL VBR
01600
01700 C ALLOCATE NEW BUFFER.
01800 IDPTR=INITOP(ISPTR,1,'HEAD EDGES')
01900
02000 C INITIALIZE FAIL DPY BUFFER.
02100 CALL IFAILB(IDPTR)
02200
02300 C ZERO NEW BUFFER.
02400 CALL ZROBUF(IDPTR)
02500 CALL IFAIL('IHEDED')
02600 CALL DELINE(-1,ISPTR,IDPTR)
02700 CALL FILCON(ISPTR,A(1))
02800 CALL INLOAD(ISPTR)
02900
03000 LLIMIT=LLINE-ILEN/5
03100
03200 C FROM LEFT, LOOK FOR VERTICAL LINE.
03300 DO 1100 I=LSIDE,RSIDE
03400 KOUNT=0
03500 DO 1100 J=FLINE,LLIMIT
03600 IF ((LOAD(I,J).AND.1).EQ.0) GO TO 1050
03700 KOUNT=KOUNT+1
03800 GO TO 1100
03900 IF (KOUNT.GE.1) GO TO 1200
04000 KOUNT=0
04100 CONTINUE
04200
04300 CONSTRAINT: THERE MUST BE A VERTICAL LINE REPRESENTING THE LEFT
04400 SIDE OF THE HEAD.
04500 IF NOT, FAILURE,
04600
04700 CALL FAIL('1100')
04800
04900 C FOUND: LEFT SIDE OF HEAD.
05000 1200 LSX=1
05100 LSYT=J-KOUNT
05200 LSYB=J-1

```



```

25320 VLST=LESTAB(LSX,LSYT)
25420 V=VLST
25520 DO 1220 J=LSYT,LSYB
25620 IF (J,NE,LSYT) V=LLINS(V,LSX,J)
25720 CALL STORE(LSX,J,LOAD(LSX,J))
25820 VLSB=V
25920
26020 C FROM RIGHT, LOOK FOR VERTICAL LINE.
26120 DO 1300 I=RSIDE,LSX,-1
26220 KOUNT=0
26320 DO 1300 J=FLINE,LLIMIT
26420 IF ((LOAD(I,J),AND.1).EQ.0) GO TO 1250
26520 KOUNT=KOUNT+1
26620 GO TO 1300
26720 IF (KOUNT,GE.1) GO TO 1400
26820 KOUNT=0
26920 CONTINUE
27020 C-----
27120 C THERE MUST BE A VERTICAL LINE REPRESENTING THE RIGHT
27220 C SIDE OF THE HEAD.
27320 C IF NOT, FAILURE,
27420 C-----
27520 CALL FAIL('1300')
27620
27720 C FOUND: RIGHT SIDE OF HEAD,
27820 RSX=I
27920 RSYT=J-KOUNT
28020 RSYB=J-1
28120 VRST=LESTAB(RSX,RSYT)
28220 V=VRST
28320 DO 1420 J=RSYT,RSYB
28420 IF (J,NE,RSYT) V=LRINS(V,RSX,J)
28520 CALL STORE(RSX,J,LOAD(RSX,J))
28620 VRSB=V
28720
28820
28920
29020
29120 C-----
29220 C THE LEFT AND RIGHT SIDES OF THE HEAD MUST BE SEPARATED.
29320 C IF NOT, FAILURE.
29420 C-----
29520 C ARE THEY LEFT AND RIGHT?
29620 IF (RSX-LSX,LT.4) CALL FAIL('1500')
29720
29820
29920 C FROM TOP, BETWEEN LSX AND RSX, LOOK FOR
30020 C HORIZONTAL LINE.
30120 DO 1600 J=FLINE,LSYT
30220 KOUNT=0
30320 DO 1600 I=LSX,RSX
30420 IF ((LOAD(I,J),AND.2).EQ.0) GO TO 1550
30520 KOUNT=KOUNT+1
30620 GO TO 1600
30720 IF (KOUNT,GE.1) GO TO 1700
30820 KOUNT=0
30920
31020
31120
31220
31320
31420
31520
31620
31720
31820
31920
32020
32120
32220
32320
32420
32520
32620
32720
32820
32920
33020
33120
33220
33320
33420
33520
33620
33720
33820
33920
34020
34120
34220
34320
34420
34520
34620
34720
34820
34920
35020
35120
35220
35320
35420
35520
35620
35720
35820
35920
36020
36120
36220
36320
36420
36520
36620
36720
36820
36920
37020
37120
37220
37320
37420
37520
37620
37720
37820
37920
38020
38120
38220
38320
38420
38520
38620
38720
38820
38920
39020
39120
39220
39320
39420
39520
39620
39720
39820
39920
40020
40120
40220
40320
40420
40520
40620
40720
40820
40920
41020
41120
41220
41320
41420
41520
41620
41720
41820
41920
42020
42120
42220
42320
42420
42520
42620
42720
42820
42920
43020
43120
43220
43320
43420
43520
43620
43720
43820
43920
44020
44120
44220
44320
44420
44520
44620
44720
44820
44920
45020
45120
45220
45320
45420
45520
45620
45720
45820
45920
46020
46120
46220
46320
46420
46520
46620
46720
46820
46920
47020
47120
47220
47320
47420
47520
47620
47720
47820
47920
48020
48120
48220
48320
48420
48520
48620
48720
48820
48920
49020
49120
49220
49320
49420
49520
49620
49720
49820
49920
50020

```



```

10500 1600 CONTINUE
10600 C-----
10700 CONSTRAINT:  THERE MUST BE A HORIZONTAL LINE REPRESENTING THE TOP
10800 C             OF THE HEAD.
10900 C             IF NOT, FAILURE.
11000 C-----
11100 CALL FAIL('1600')
11200
11300 C             FOUND:  TOP OF HEAD
11400 1700 TLX=I-KOUNT
11500 TRX=I-1
11600 TY=J
11700 VTL=LESTAB(TLX,TY)
11800 V=VTL
11900 DO 1720 I=TLX,TRX
12000 IF (I,NE,TLX) V=LIRINS(V,I,TY)
12100 CALL STORE(I,TY,LOAD(I,TY))
12200 VTR=V
12300
12400 C-----
12500 CONSTRAINT:  THE TOP OF THE HEAD MUST BE ABOVE THE SIDES OF THE
12600 C             HEAD.
12700 C             IF NOT, THE SIDE(S) IS(ARE) IN ERROR.  DELETE SIDE(S)
12800 C             AND TRY AGAIN.
12900 C-----
13000 C             KABOVE CHANGED FROM 3 TO 2 ON 8 JUN 70 WHILE DEBUGGING RPHH1
13100 C             KABOVE=2
13200 IF (LSYT-TY,GT,KABOVE) GO TO 17320
13300 CALL NFATAL
13400 CALL FAIL('1732')
13500 CALL YFATAL
13600 CALL DELINE(0,VLSB,VLST)
13700 IF (RSYT-TY,GT,KABOVE) GO TO 17330
13800 GO TO 17325
13900 17320 IF (RSYT-TY,GT,KABOVE) GO TO 17340
14000 17325 CALL NFATAL
14100 CALL FAIL('1732')
14200 CALL YFATAL
14300 CALL DELINE(0,VIRST,VRSB)
14400 CALL KILLLL
14500 GO TO 50
14600
14700 C-----
14800 CONSTRAINT:  THE TOP OF THE HEAD MUST BE BETWEEN THE SIDES OF THE
14900 C             HEAD.
15000 C             IF NOT, THE TOP IS IN ERROR.  DELETE THE TOP AND
15100 C             TRY AGAIN.
15200 C-----
15300 17340 IF (LSX,LT,TLX-2,AND,RSX,GT,TRX+2) GO TO 1738
15400 CALL NFATAL
15500 CALL FAIL('1734')
15600 CALL YFATAL
15700

```



```

14700 CALL DELINE(0,VTL,VTR)
14800 CALL KILLLL
14900 GO TO 50
15000
15100 C ASSERT: TENTATIVE SIDES AND TOP OF HEAD HAVE BEEN LOCATED.
15200 C LEFT LSX,LSYT,LSYB VLSB,VLSB
15300 C RIGHT RSX,RSYT,RSYB VRST,VRST
15400 C TOP TLX,TRX, TY VTL,VTR
15500 C SEARCH FOR LONG, FAIRLY STRAIGHT, SIDES OF THE HEAD.
15600 C LONG AND FAIRLY STRAIGHT ARE DEFINED IN TERMS OF
15700 C HEAD WIDTH AND Y COORDINATE OF TOP.
15800 C HEDWID=RSX-LSX
15900 C FAIRLY STRAIGHT: MUST BE IN BAND FSBAND WIDE.
16000 C FSBAND=2*(HEDWID/10)
16100 C ININC=FSBAND/2
16200 C OUTINC=FSBAND-ININC-1
16300 C LONG: SEARCH BAND FROM BNDTOP TO BNDBOT.
16400 C BNDBOT=TY+HEDWID-FSBAND
16500 C BNDTOP=TY+FSBAND
16600 C-----
16700 C CONSTRAINT: BAND MUST BE ENTIRELY WITHIN PICTURE.
16800 C IF NOT, SIDE CLOSEST TO EDGE IS IN ERROR. DELETE
16900 C IT AND TRY AGAIN.
17000 C-----
17100 C IF (BNDBOT,LT,LL,LINE-2) GO TO 1740
17200 CALL NFATAL
17300 CALL FAIL('1739')
17400 CALL YFATAL
17500 LOIST=LSX-LSIDE
17600 ROIST=RSIDE-RSX
17700 IF (LOIST-ROIST) 1830,1870,1870
17800
17900 RIGHT=1
18000 LEFT=-1
18100 C LEFT TOP.
18200 IF (BNDTOP,GE,LSYT) GO TO 1741
18300 CALL SIDSRC(VLST,RIGHT,LSX-OUTINC,LSX+ININC,LSYT-1,BNDTOP)
18400 LSYT=LYCOR(VLST)
18500 C RIGHT TOP.
18600 IF (BNDTOP,GE,RSYT) GO TO 1742
18700 CALL SIDSRC(VRST,LEFT,RSX+OUTINC,RSX-ININC,RSYT-1,BNDTOP)
18800 RSYT=LYCOR(VRST)
18900 C LEFT BOTTOM.
19000 IF (BNDBOT,LE,LSYB) GO TO 1743
19100 CALL SIDSRC(VLSB,LEFT,LSX-OUTINC,LSX+ININC,LSYB+1,BNDBOT)
19200 LSYB=LYCOR(VLSB)
19300 C RIGHT BOTTOM.
19400 IF (BNDBOT,LE,RSYB) GO TO 1758
19500 CALL SIDSRC(VRSB,RIGHT,RSX+OUTINC,RSX-ININC,RSYB+1,BNDBOT)
19600 RSYB=LYCOR(VRSB)
19700 C FOLLOW EDGES FROM SIDE TO TOP OF HEAD.
19800 C

```



```

19000  V=VLST
20000  CALL UHEF(1,LXCOR(VLST),LSYT,TLX,TY,V,FAILSW)
20100  IF (FAILSW,EQ,0) GO TO 1760
20200  -----
20300  C UHEF FAILED. LEFT SIDE OF HEAD MUST BE IN ERROR.
20400  C REMOVE "LEFT SIDE LINE" FROM SOURCE PICTURE AND TRY AGAIN.
20500  C -----
20600  GO TO 1830
20700  -----
20800  1760  CALL LLINK(V,VTL)
20900  V=VRST
21000  CALL UHEF(-1,LXCOR(VRST),RSYT,TRX,TY,V,FAILSW)
21100  IF (FAILSW,EQ,0) GO TO 1780
21200  -----
21300  C UHEF FAILED. RIGHT SIDE OF HEAD MUST BE IN ERROR.
21400  C REMOVE "RIGHT SIDE LINE" FROM SOURCE PICTURE AND TRY AGAIN.
21500  C -----
21600  GO TO 1870
21700  -----
21800  1780  CALL LLINK(VTR,V)
21900  C FOLLOW EDGES FROM SIDE TO BOTTOM OF PICTURE.
22000  V=VLSB
22100  CALL LHEF(1,LXCOR(VLSB),LSYB,LLINE,V,FAILSW)
22200  VBL=V
22300  IF (FAILSW,EQ,0) GO TO 1850
22400  -----
22500  C LHEF FAILED. LEFT SIDE OF HEAD MUST BE IN ERROR.
22600  C REMOVE "LEFT SIDE LINE" FROM SOURCE PICTURE AND TRY AGAIN.
22700  C -----
22800  1830  CALL DELINE(0,VLSB,VLST)
22900  CALL KILLLL
23000  GO TO 50
23100  -----
23200  1850  V=VRSB
23300  CALL LHEF(-1,LXCOR(VRSB),RSYB,LLINE,V,FAILSW)
23400  VBR=V
23500  IF (FAILSW,EQ,0) GO TO 1900
23600  -----
23700  C LHEF FAILED. DO AS ABOVE.
23800  C -----
23900  1870  CALL DELINE(0,VRST,VRSB)
24000  CALL KILLLL
24100  GO TO 50
24200  -----
24300  C REMOVE CONCAVITIES IN SIDES.
24400  1900  CALL REMOCC(VLSB,VLST,1)
24500  CALL REMOCC(VRST,VRSB,-1)
24600  -----
24700  C RELEASE FAIL DPY BUFFER.
24800  CALL RFAILB
24900  IHEDED=IDPTR
25000  -----

```


PAGE 3-6

1,MDK

BOR34

0833

END

17-JUL-70

25100


```

00100 SUBROUTINE SIDSRC(VPTR,LORR,XSTART,XEND,YSTART,YEND)
00200 IMPLICIT INTEGER (A-Z)
00300 C SIDE SEARCH.
00400
00500 XSTEP=1
00600 IF (XEND.LT.XSTART) XSTEP=-1
00700 YSTEP=1
00800 IF (YEND.LT.YSTART) YSTEP=-1
00900 DO 200 J=YSTART,YEND,YSTEP
01000 DO 100 I=XSTART,XEND,XSTEP
01100 L=LOAD(I,J)
01200 IF ((L.AND.13).NE.0) GO TO 150
01300 CONTINUE
01400 GO TO 200
01500 VPTR=LPINS(LORR,VPTR,I,J)
01600 CALL STORE(I,J,L)
01700 CONTINUE
01800 RETURN
01900 END
02000
02100
02200
02300
02400
02500
02600
02700
02800
02900
03000
03100
03200
03300
03400
03500
03600
03700
03800
03900
04000
04100
04200
04300
04400
04500
04600
04700
04800
04900
05000
05100
05200

```

SUBROUTINE REMOCC(V1,V2,WHERE)
IMPLICIT INTEGER (A-Z)
V=V1
X1=LXCOR(V)
V=LRSIB(V)
IF (V.EQ.V2) RETURN
X2=LXCOR(V)
IF ((X2-X1)*WHERE) 30,30,40
X1=X2
GO TO 20
X2 IS INWARD OF X1.
K=1
V=LRSIB(V)
IF (V.EQ.V2) RETURN
X2=LXCOR(V)
IF ((X2-X1)*WHERE) 100,100,60
STILL INWARD.
IF (K,GT.4) GO TO 10
K=K+1
GO TO 50
FIX UP K OF THEM.
W=V
W=LRSIB(W)
TX=LXCOR(W)
TY=LXCOR(W)
CALL LSETXY(W,TX-WHERE,TY)
CALL STORE(TX,TY,0)
CALL STORE(TX-WHERE,TY,15)
K=K-1
IF (K,GT.0) GO TO 110

GO TO 10
END

05300
05400


```

00100 SUBROUTINE DELINE(SW,P1,P2)
00200 IMPLICIT INTEGER(A-Z)
00300
00400 IF (SW) 100,200,200
00500
00600 SW=-1, P1 AND P2 ARE PICTURE POINTERS. THIS IS AN
00700 INITIALIZATION CALL.
00800 PTR1=P1
00900 PTR2=P2
01000 RETURN
01100
01200 SW=0 P1 AND P2 ARE LIST POINTERS, P2 ON RIGHT.
01300 DELETE POINTS IN PICTURE WHICH ARE ON LIST, INCLUDING
01400 END POINTS.
01500 CALL INSTOR(PTR1)
01600 J=P1
01700 X=LYCOR(J)
01800 Y=LYCOR(J)
01900 CALL STORE(X,Y,0)
02000 IF (J.EQ.P2) GO TO 300
02100 J=LRSIB(J)
02200 GO TO 250
02300 CALL INSTOR(PTR2)
02400 RETURN
02500
02600 END

```



```

20100 SUBROUTINE UHEF(WHERE,SX,SY,FX,LY,LY,V,FAILSW)
20200 IMPLICIT INTEGER (A-Z)
20300 UPPER HEAD EDGE FOLLOWER.
20400 FOLLOWS A CURVING EDGE FROM THE TOP OF THE SIDE OF
20500 THE HEAD (SX,SY) TO THE TOP OF THE HEAD (TX,TY).
20600 WHERE = 1 : UPPER LEFT CURVE OF HEAD.
20700 WHERE = -1 : UPPER RIGHT CURVE OF HEAD.
20800 [BRACKETED COMMENTS APPLY TO WHERE=-1,3]
20900 COMMON /CON/A(3),BITS,INID,LINLEN,FLINE,LLINE,LSIDE,RSIDE,
21000 1 TVPNT,TVSIZ,NPERND,ILEN,B(8),ITEXT(10)
21100
21200 CALL NFATAL
21300 FX=SX
21400 FY=SY
21500 FD=LOAD(FX,FY)
21600 IF (WHERE,LT,0) GO TO 120
21700 OKDIRS=1+2+4
21800 OKSIDE=1+4
21900 OKTOPS=2+4
22000 GO TO 300
22100 OKDIRS=1+2+8
22200 OKSIDE=1+8
22300 OKTOPS=2+8
22400
22500 FX = LAST X FOUND
22600 FY = LAST Y FOUND
22700 FD = DIRECTION AT FX,FY
22800
22900 IF FY=TY, TOP HAS BEEN REACHED.
23000 IF (FY,LE,TY) GO TO 400
23100
23200 DOES FD INCLUDE VERTICAL EDGE?
23300 IF ((FD,AND,1),EQ,0) GO TO 3500
23400
23500 YES.
23600 DO 3100 KOUNT=1,5
23700 IF (FY-KOUNT,LT,TY) GO TO 3200
23800 IF (KOUNT,GT,1) GO TO 3070
23900 ILIM=FX+WHERE
24000 GO TO 3075
24100 ILIM=FX+2*WHERE
24200 DO 3100 I=FX-WHERE,ILIM,WHERE
24300 IF (I,LT,LSIDE,OR,I,GT,RSIDE) GO TO 3100
24400 P=LOAD(I,FY-KOUNT)
24500 I OR ? C I OR ? J
24600 IF ((P,AND,OKSIDE),NE,0) GO TO 3800
24700 CONTINUE
24800 CALL FAIL('3100')
24900 CANNOT FIND EDGE ABOVE FX,FY. TRY FOR EDGE AS IF FD WAS A SLANT.
25000 GO TO 3500
25100 CALL FAIL('3020')
25200 GO TO 5300

```



```

05300 C
05400 C
05500 C
05600 3500
05700
05800
05900 C
06000 C
06100 C
06200 3520
06300
06400 C
06500
06600
06700
06800 3530
06900
07000 C
07100 C
07200
07300
07400 3600
07500
07600
07700 3700
07800
07900
08000 C
08100 3800
08200
08300
08400
08500
08600
08700
08800
08900 C
09000 C
09100
09200 4000
09300
09400
09500
09600
09700
09800
09900
10000 4100
10100 C
10200
10300
10400

NO,
THEREFORE FD HAS / OR - EDGE. [ \ OR - EDGE ]
I3500=0
DO 3600 KOUNT=1,5
IF (FY-KOUNT-TY) 3700,3520,3520
SEARCH FROM VERTICALLY ABOVE FX,FY INWARD TO
LINE BETWEEN FX,FY AND TX,TY.
RATIOS: NRSTEPS / COUNT = TX-FX / FY-TY
NNUM=(TX-FX)*WHERE*KOUNT
NDEN=FY-TY
NRSTEP=NNUM/NDEN
ROUND UP.
IF (MOD(NNUM,NDEN).NE.0) NRSTEP=NRSTEP+1
I3530=FX+NRSTEP*WHERE
DO 3600 I=FX,I3530,WHERE
IF (I.LT.LSIDE.OR.I.GT.RSIDE) GO TO 3600
P=LOAD(I,FY-KOUNT)
I OR / OR -? (I.e., ANYTHING BUT \ ONLY.)
[ \ ]
[ / ]
IF ((P.AND.OKDIRS).NE.0) GO TO 3800
CONTINUE
CALL FAIL('3600')
GO TO 5300
CALL FAIL('3520')
GO TO 5300

FOUND
FX=I
FY=FY-KOUNT
FD=P
IF (FX.EQ.TX.AND.FY.EQ.TY) GO TO 5000
CALL STORE(FX,FY,P)
VLPINS(WHERE,V,FX,FY)
GO TO 3000

TOP HAS BEEN REACHED. INCLUDE ANY - OR / [ - OR \ ]
OVER TO PREVIOUS TOP LINE.
DO 4100 I=FX+WHERE,FX+3*WHERE,WHERE
IF ((TX-I)*WHERE.LE.0) GO TO 5000
P=LOAD(I,TY)
IF ((P.AND.OKTOPS).EQ.0) GO TO 4100
CALL STORE(I,TY,P)
VLPINS(WHERE,V,FX,FY)
FX=I
GO TO 4000
CONTINUE
A GAP OF MORE THAN TWO POINTS HAS BEEN FOUND
IN THE TOP OF THE HEAD. FAILURE.
CALL FAIL('4120')
GO TO 5300

```


10520	5000	FAILSW=0
10620		GO TO 5600
10720	5300	FAILSW=1
10820	5600	CALL YFATAL
10920		RETURN
11000		END

Digitized by Google


```

05322 C      If new is close to vertical starting line continue
05400 C      with Stage 1.
05520 6115 IF (FAILCM,EQ.1) GO TO 6990
05600 IF (FY,GE.8Y) GO TO 6150
05700 IF ((FX-SAVE)*WHERE,LT.1) GO TO 6000
05800 NEXT STMT NICE FOR KOCT2. IS IT GENERALLY USEFUL TO
05900 C      OMIT STAGE 2?
06000 C      GO TO 6300
06032 C      8 JUNE 70 - NO FOR LOEH2
06060 GO TO 6200
06100
06200 6150 CALL FAIL('6150')
06300 GO TO 6990
06400
06500 C #####
06620 C      Stage 2, going down.
06720 6200 IF (FY,GE.8Y) GO TO 7000
06820 IF ((FD,AND,SLNOUT),EQ.0) GO TO 6210
06900 STAGE=2
07022 GO TO 6303
07120 6210 CALL SEARCH(FX-WHERE,FX+WHERE,WHERE,1+4+8,'6205',-1)
07220 C      OK, new edge.
07320 C      If not back under vertical edge, continue with Stage 2.
07420 IF (FAILCM,EQ.1) GO TO 6990
07520 IF ((FX-SAVE)*WHERE,GT.0) GO TO 6200
07620
07720 C #####
07820 C      Stage 3, going out.
07920 6300 IF (FY,GE.8Y) GO TO 7000
08022 STAGE=3
08120 C      Does FD include a slant edge?
08220 IF ((FD,AND,SLNOUT),EQ.0) GO TO 6310
08320 C      Yes.
08420 C      IF POINT BELOW IS NOT A 2,
08520 C      FIRST TRY TO FOLLOW OUTWARD.
08620 6303 TFD=LOAD(FX,FY+1)
08720 IF ((TFD,AND,2).NE.0) GO TO 6307
08820 TFD=LOAD(FX-WHERE,FY)
08920 IF ((TFD,AND,DWNOUT),EQ.0) GO TO 6307
09020 FX=FX-WHERE
09100 FD=TFD
09200 CALL STORE(FX,FY,FD)
09300 VCM=LPINS(L OR R,VCM,FX,FY)
09420 IF (STAGE,EQ.2) GO TO 6200
09500 GO TO 6300
09620 TEMPFX=FX
09700 6307 CALL SEARCH(FX-2*WHERE,TEMPFX,WHERE,DWNOUT,'6305',-1)
09820 GO TO 6315
09900 C      No. FD is 1.
10000 6310 CALL SEARCH(FX-2*WHERE,FX+WHERE,WHERE,DWNOUT,'6310',-1)
10120 6315 IF (FAILCM,EQ.1) GO TO 6990
10220 IF (STAGE,EQ.2) GO TO 6200

```


10300	GO TO 6300
10400	
10500	FAILSW=1
10600	GO TO 7010
10700	FAILSW=0
10800	CALL YFATAL
10900	V=VCM
11000	RETURN
11100	END


```

00100 SUBROUTINE SEARCH(FROMP,TO,STEP,DIRS,FLABEL,WIDEN)
00200 IMPLICIT INTEGER (A-Z)
00300 COMMON /LHEFCM/ BY,FX,FY,FD,FAILCM,VCM,L OR R
00400 FROM=FROMP
00500 DO 120 KOUNT=1,5
00600 FY=FY+1
00700 IF (FY.GT.BY) RETURN
00800 DO 100 FX=FROM,TO,STEP
00900 FD=LOAD(FX,FY)
01000 IF ((FD.AND.DIRS).NE.0) GO TO 200
01100 CONTINUE
01200 IF (WIDEN.NE.0) FROM=FROM+STEP
01300 CONTINUE
01400 CALL FAIL(FLABEL)
01500 FAILCM=1
01600 RETURN
01700
0200 CALL STORE(FX,FY,FD)
VCM=LPINS(L OR R,VCM,FX,FY)
END
22000

```



```

00120 SUBROUTINE DSLINS(PTR,X,Y,POG)
00200 IMPLICIT INTEGER (A-Z)
00300 COMMON /CON/ A(3),BITS,INID,LINLEN,FLINE,LLINE,LSIDE,RSIDE,
00400 1 TVPNTR,TVSIZE,NPERWD,ILEN,8(8),ITEXT(10)
00500
00600 CALL FILCON(PTR,A(1))
00700 CALL INLOAD(PTR)
00800
00900 DO 500 J=FLINE,LLINE
01000 DO 500 I=LSIDE,RSIDE
01100 P=LOAD(I,J)
01200 IF (P.EQ.0) GO TO 5000
01300
01400 1
01500 IF (MOD(P,2).NE.1) GO TO 3200
01600 IF (J.EQ.FLINE) GO TO 3110
01700 IF (MOD(LOAD(I,J-1),2).EQ.1) GO TO 3200
01800 IF (J.EQ.LLINE) GO TO 3125
01900 DO 3120 JJ=J+1,LLINE
02000 IF (MOD(LOAD(I,JJ),2).NE.1) GO TO 3130
02100 CONTINUE
02200 JJ=LLINE+1
02300 JJ=JJ-1
02400 CALL ALINE(X+6*(I-LSIDE),Y+3-6*(J-FLINE),
02500 1 X+6*(I-LSIDE),Y-3-6*(JJ-FLINE) )
02600
02700 2
02800 IF (MOD(P/2,2).NE.1) GO TO 3300
02900 IF (I.EQ.LSIDE) GO TO 3210
03000 IF (MOD(LOAD(I-1,J)/2,2).EQ.1) GO TO 3300
03100 IF (I.EQ.RSIDE) GO TO 3225
03200 DO 3220 II=I+1,RSIDE
03300 IF (MOD(LOAD(II,J)/2,2).NE.1) GO TO 3230
03400 CONTINUE
03500 II=RSIDE+1
03600 II=II-1
03700 CALL ALINE(X+3+6*(I-LSIDE),Y-6*(J-FLINE),
03800 1 X+3+6*(II-LSIDE),Y-6*(J-FLINE) )
03900
04000 4
04100 IF (MOD(P/4,2).NE.1) GO TO 3400
04200 IF (I.EQ.RSIDE.OR.J.EQ.FLINE) GO TO 3310
04300 IF (MOD(LOAD(I+1,J-1)/4,2).EQ.1) GO TO 3400
04400 KK=MIN0(I-LSIDE,LLINE-J)
04500 IF (KK.EQ.0) GO TO 3325
04600 DO 3320 K=1,KK
04700 IF (MOD(LOAD(I-K,J+K)/4,2).NE.1) GO TO 3330
04800 CONTINUE
04900 K=KK+1
05000 K=K-1
05100 CALL ALINE(X+3+6*(I-LSIDE),Y+3-6*(J-FLINE),
05200 1 X-3+6*(I-K-LSIDE),Y-3-6*(J+K-FLINE))

```


17-JUL-70	0833	BOR34	1,MDK
05300			
05400	C		
05500	3400		
05600			
05700			
05800	3410		
05900			
06000			
06100			
06200	3420		
06300	3425		
06400	3430		
06500			
06600			
06700			
06800	5000		
06900	5500		
07000			
07100			

```

\      8
IF (MOD(P/8,2).NE.1) GO TO 5000
IF (I.EQ.LSIDE.OR.J.EQ.FLINE) GO TO 3410
IF (MOD(LOAD(I-1,J-1)/8,2).EQ.1) GO TO 5000
KK=MIN0(RSIDE-I,LLINE-J)
IF (KK.EQ.0) GO TO 3425
DO 3420 K=1,KK
IF (MOD(LOAD(I+K,J+K)/8,2).NE.1) GO TO 3430
CONTINUE
K=KK+1
K=K-1
CALL ALINE(X-3+6*(I-LSIDE),Y-3-6*(J-FLINE),
1      X+3+6*(I+K-LSIDE),Y-3-6*(J+K-FLINE))
CONTINUE
CALL DPYOUT(POG)
END

```



```

001200 FUNCTION JRIGHE(RIGPT,HEDPOG)
002000 INTEGER BIGPT,HEDPOG
003000 COMMON /CON/ A(3),BITS,INID,LINLEN,FLINE,LLINE,LSIDE,RSIDE,
004000 1 TVPNTR,TVSIZ,NPERWD,ILEN,B(8),ITEXT(10)
005000 INTEGER A,BITS,FLINE,RSIDE,TVPNTR,TVSIZ,B
006000 COMMON /LENTS/ VBL,VLSB,VLST,VTL,VTR,VRST,VRSB,VBR
007000 INTEGER VBL,VLSB,VLST,VTL,VTR,VRST,VRSB,VBR
008000 COMMON /FOLCOM/ IP1,JP1,IP2,JP2,IP3,JP3,IQ,JQ,MPOG,
009000 1 WSTART,WFIN
010000 INTEGER WSTART,WFIN
011000 EXTERNAL JEDGES
012000
012005 C
012050 C
012070 C
012080 C
012100 C
012110 C
012150 C
012200 C
012250 C
012300 C
012350 C
013000 C
013020 C
014000 C
015000 C
016000 C
017000 C
018000 C
019000 C
020000 C
021000 C
022000 C
023000 C
024000 C
025000 C
026000 C
027000 C
028000 C
029000 C
030000 C
031000 C
032000 C
033000 C
034000 C
035000 C
036000 C
037000 C
038000 C
039000 C
040000 C
041000 C
042000 C
043000 C

FUNCTION JRIGHE(RIGPT,HEDPOG)
INTEGER BIGPT,HEDPOG
COMMON /CON/ A(3),BITS,INID,LINLEN,FLINE,LLINE,LSIDE,RSIDE,
1 TVPNTR,TVSIZ,NPERWD,ILEN,B(8),ITEXT(10)
INTEGER A,BITS,FLINE,RSIDE,TVPNTR,TVSIZ,B
COMMON /LENTS/ VBL,VLSB,VLST,VTL,VTR,VRST,VRSB,VBR
INTEGER VBL,VLSB,VLST,VTL,VTR,VRST,VRSB,VBR
COMMON /FOLCOM/ IP1,JP1,IP2,JP2,IP3,JP3,IQ,JQ,MPOG,
1 WSTART,WFIN
INTEGER WSTART,WFIN
EXTERNAL JEDGES

MAPPING FROM REDUCED POINTS TO BIG POINTS.
*** BEWARE *** THESE ASP'S ARE IMPLICITLY IN IVFIX AT
STATEMENTS 40 AND 140.

IMAP(I)=8*I+LSIDE
JMAP(J)=8*J+FLINE
OLD MAPPING.
IMAP(I)=8*I+LSIDE-4
JMAP(J)=8*J+FLINE-4

ALLOCATE AND ZERO NEW BUFFER.
NEWPTR=INITOP(BIGPT,1,'BIG EDGES')
CALL ZROBUF(NEWPTR)

INITIALIZE DPY STUFF
MPOG=HEDPOG
CALL FOLLOW(1,NUTN)

START PROCESSING
LIN=VBL
IL1=LXCOR(LIN)
JL1=LYCOR(LIN)
IP1=IMAP(IL1)
JP1=JMAP(JL1)
IQ=IP1
JQ=JP1
LIN=LRSIB(LIN)
IL2=LXCOR(LIN)
JL2=LYCOR(LIN)
RTEMP=IL2-IL1
IF (RTEMP) 120,110,120
SLOP12=999.
GO TO 130
SLOP12=(JL2-JL1)/RTEMP
LIN=LRSIB(LIN)
IF (LIN.EQ.0) GO TO 180
IL3=LXCOR(LIN)
JL3=LYCOR(LIN)
RTEMP=IL3-IL2
IF (RTEMP) 150,140,150
SLOP23=999.

```



```

04420      GO TO 160
04520      SLOP23=(JL3-JL2)/RTEMP
04620      IF (SLOP12-SLOP23) 200,170,200
04722      C      SLOPES ARE EQUAL
04800      170      IL2=IL3
04923      JL2=JL3
05020      GO TO 130
05123      C
05200      END OF LIST. SET UP DUMMY P3.
05322      180      IP2=IMAP(IL2)
05420      JP2=JMAP(JL2)
05522      IP3=2*IP2-IP1
05623      JP3=2*JP2-JP1
05720      GO TO 250
05820      C
05923      SLOPES NOT EQUAL. P1,P2, AND P3 FOUND.
06020      200      IP2=IMAP(IL2)
06120      JP2=JMAP(JL2)
06220      IP3=IMAP(IL3)
06323      JP3=JMAP(JL3)
06420      C
06520      NOW PROCESS SECTION FROM P1 TO P2.
06622      250      CALL FOLLOW(0,JEDGE5)
06720      C
06800      SECTION PROCESSED. ON TO NEXT. DUM-DI-DUMP-DUMP.
06920      IF (LIN.EQ.0) GO TO 400
07020      SLOP12=SLOP23
07123      IP1=IP2
07223      JP1=JP2
07320      GO TO 170
07420      C
07500      ALL FINISHED.
07622      400      CALL FOLLOW(-1,NUTN)
07722      JBICHE=NEWPTR
07820      C
07923      PUT NEW LIST ENTRIES IN.
08020      VBL=HSTART
08123      VLSB=IVFIX(VBL,VLSB,1)
08203      VLSI=IVFIX(VLSB,VLSI,1)
08302      VTL=IVFIX(VLSI,VTL,-2)
08400      VTR=IVFIX(VTL,VTR,-2)
08522      VRST=IVFIX(VTR,VRST,-1)
08623      VRSB=IVFIX(VRST,VRSB,-1)
08720      VBR=WFIN
08820      RETURN
08920      END
09000

```


17-JUL-70	0840	HEAD26	1,MDK	PAGE 2-1
00120		FUNCTION IVFIX(V1,V2,ISIGN)		
00200		IMPLICIT INTEGER (A-Z)		
00300	C	IISIGNI = 1 : TEST Y		
00400	C	IISIGNI = 2 : TEST X		
00500		COMMON /CON/ A(3),BITS,IWID,LINLEN,FLINE,LLINE,LSIDE,RSIDE,		
00600		1 TVPNTR,TVSIZ,NSPERWD,ILEN,B(8),ITEXT(10)		
00700		INTEGER A,RITS,FLINE,RSIDE,TVPNTR,TVSIZ,B		
00800		IF (IARS(ISIGN),EQ,1) GO TO 140		
00850	C	*** BEWARE *** NEXT STMT IS ASF IMAP FROM JBIGHE.		
00900	40	I=LSIDE+8*LXCOR(V2)		
01000		J=V1		
01100	100	J=LRSIB(J)		
01200		ITEST=LXCOR(J)		
01300		IF (ISIGN*(I-ITEST)) 100,200,200		
01350	C	*** BEWARE *** NEXT STMT IS ASF JMAP FROM JBIGHE.		
01400	140	I=FLINE+8*LYCOR(V2)		
01500		J=V1		
01600	160	J=LRSIB(J)		
01700		ITEST=LYCOR(J)		
01800		IF (ISIGN*(I-ITEST)) 160,200,200		
01900	200	IVFIX=J		
02000		END		


```

00100      FUNCTION MSD(SLOPE)
00200      C      MAIN SLANT DIRECTION.
00300      IF (SLOPE) 1140,1110,1120
00400      MDIRS=2
00500      C      ORIGINALLY 14
00600      GO TO 1200
00700      1120      IF (SLOPE-1.) 1135,1130,1133
00800      1130      MDIRS=4
00900      C      ORIGINALLY 7
01000      GO TO 1200
01100      MDIRS=5
01200      GO TO 1200
01300      MDIRS=6
01400      GO TO 1200
01500      1140      IF (SLOPE+1.) 1155,1150,1153
01600      1150      MDIRS=8
01700      C      ORIGINALLY 11
01800      GO TO 1200
01900      MDIRS=10
02000      GO TO 1200
02100      1155      MDIRS=9
02200      GO TO 1200
02300      MSD=MDIRS
02400      END

```



```

00120 SUBROUTINE FOLLOW(ISW,JFUNC)
00220 COMMON /FOLCOM/ IP1,JP1,IP2,JP2,IP3,JP3,IQ,JQ,MPOG,
00320 1 WSTART,WFIN
00420 INTEGER WSTART,WFIN
00520 DPY STUFF -----
00620 INTEGER G1,G2
00700 REAL GCONST
00820 DATA GCONST/6.0/
00900 INTEGER TPOG,UPOG,SPOG,SFLAG
01000 DATA LSBUF/40/
01120 LOGICAL NOPDY,ANYM,ANYN,ANYS
01200 DATA NOPDY,TRUE./
01300 PIECE OF GLASS
01400 TPOG
01500 UPOG
01620 SPOG
01720 MPOG
01820 NPOG
01920 -----
02000 LOGICAL HORIZ,WIDE,NXTWID,ANYLIS
02100 DIMENSION II(-4/11),JJ(-4/11),ISTOP(-4/11),JSTOP(-4/11)
02200 DATA NB1/-4/,NB2/11/
02320 COMMON /OUTFGS/ IFDTTY,IFOLPT,IFDDPY
02400
02500 IF (ISW) 400,1000,100
02620
02700 INITIALIZE
02820 IF (IFDDPY.EQ.0) NOPDY=,TRUE.
02920 ANYM=,FALSE.
03020 SFLAG=0
03100 ANYS=,FALSE.
03200 WIDE=,TRUE.
03320 ANYLIS=,FALSE.
03400 IF (NOPDY) RETURN
03500 TPOG=IADPST(25)
03620 UPOG=IADPST(50)
03700 NPOG=IADPST(200)
03820 RETURN
03900
04000 ALL FINISHED.
04100 WFIN=JLISPT
04200 IF (IFDDPY.EQ.0) RETURN
04320 IF (NOPDY) RETURN
04420 CALL OUTD('++DIS OK?&')
04520 NUTN=YORN(-1)
04620 CALL HYPOG(TPOG)
04702 CALL HYPOG(UPOG)
04820 CALL HYPOG(NPOG)
04920 IF (,NOT,ANYS) GO TO 430
05000 CALL HYPOG(SPOG)
05120 CALL IGIVEX(SPOG)
05200 CALL IGIVEX(NPOG)

```



```

05300 CALL IGIVEX(UPOG)
05420 CALL IGIVEX(TPOG)
05520 RETURN
05600
05700 *****
05800 ***** PROCESS SECTION FROM P1 TO P2 *****
05900 C INPUT TO THIS PART OF CODE IS:
06000 C COORDINATES OF P1,P2,P3, AND Q WHICH ARE
06100 C IP1,JP1
06200 C IP2,JP2
06300 C IP3,JP3
06400 C IQ,JQ
06500
06600
06700
06800
06900
07000
07100
07200
07300
07400
07500
07600
07700
07800
07900
08000
08100
08200
08300
08400
08500
08600
08700
08800
08900
09000
09100
09200
09300
09400
09500
09600
09700
09800
09900
10000
10100
10200
10300
10400

C *****
C ***** PROCESS SECTION FROM P1 TO P2 *****
C INPUT TO THIS PART OF CODE IS:
C COORDINATES OF P1,P2,P3, AND Q WHICH ARE
C IP1,JP1
C IP2,JP2
C IP3,JP3
C IQ,JQ
C
C DRAW LINES CONNECTING THE 3 POINTS
C IF (NOOPY) GO TO 1010
C CALL CLRPOG(TPOG)
C CALL ALINE(G1(IP1),G2(JP1),G1(IP2),G2(JP2))
C CALL AVECT(G1(IP3),G2(JP3))
C CALL DPYOUT(TPOG)
C
C DETERMINE LINE L WHICH BISECTS ANGLE BETWEEN P1P2 AND P2P3.
C 1010
C X3=IP3-IP2
C Y3=JP3-JP2
C DIST3=SQRT(X3**2+Y3**2)
C X1=IP1-IP2
C Y1=JP1-JP2
C DIST1=SQRT(X1**2+Y1**2)
C R IS POINT ON LINE L. CONSIDERING P2R, P2P3, AND P2P1 AS
C VECTORS VR, V3, AND V1; R IS DEFINED BY:
C VR=(V3/ABS(V3)) + (V1/ABS(V1))
C [RELATIVE COORDINATES FOR POINT R ARE CALCULATED. TRUE COORDINATES
C WOULD BE IP2+XR AND JP2+YR.]
C XR=(X3/DIST3)+(X1/DIST1)
C YR=(Y3/DIST3)+(Y1/DIST1)
C
C IF (NOOPY) GO TO 1100
C DISPLAY LINE IN DIRECTION OF R OF LENGTH 8.
C DISTR=SQRT(XR**2+YR**2)
C IR8=IP2+FIX(8.*XR/DISTR+.5)
C JR8=JP2+FIX(8.*YR/DISTR+.5)
C CALL ALINE(G1(IP2),G2(JP2),G1(IR8),G2(JR8))
C CALL DPYOUT(TPOG)
C
C DETERMINE MAIN SEARCH DIRECTIONS FOR P1P2.
C 1=1 2=- 4=/ 8=\
C IF (X1,EQ,0) GO TO 1175
C SLOPE IS IN NORMAL RIGHT HANDED COORDINATE SYSTEM (HENCE MINUS).
C SLOPE=-Y1/X1
C MDIRS=MSD(SLOPE)
C GO TO 1180
C MDIRS=1
C 1175
C ORIGINALLY 13

```



```

10520
10600 C
10700 1180 DETERMINE SECONDARY SEARCH DIRECTIONS FROM P2P3.
10800 IF (X3.EQ.0) GO TO 1190
10900 EPOLS=-Y3/X3
11000 MDIRS=MDIRS.OR. MSD(EPOLS)
11100 GO TO 1200
11200 MDIRS=MDIRS.OR. 1
11300
11400 C If the search angle changes greatly (more than 45 degrees)
11500 C at P2, set flag to force a wide search at P2.
11600 C  $\cos \alpha = (-V1.V3) / (|V1|*|V3|)$ 
11700 1200 COSALF=(-X1*X3-Y1*Y3)/(DIST1*DIST3)
11800 NXTWID=.FALSE.
11900 IF (COSALF.LT.0.707) NXTWID=.TRUE.
12000
12100 C IS P1P2 MOSTLY HORIZONTAL?
12200 C IF (ABS(X1).LT.ABS(Y1)) GO TO 1400
12300 C YES
12400 C HORIZ=.TRUE.
12500 C IN GOING FROM P1 TO P2, THE X INCREMENT WILL BE +1 OR -1.
12600 C IINC=ISIGN(X1,IP2-IP1)
12700 C THE Y INCREMENT.
12800 C YINC=-Y1/ABS(X1)
12900 C THE INCREMENTS FOR SCANNING ACROSS TO LOOK FOR AN EDGE.
13000 C XSCNIN=-YINC
13100 C JSCNIN=IINC
13200 C
13300 1310 C FILL IN OFFSET TABLE FOR SCANNING
13400 C DO 1310 K=NB1,NB2
13500 C II(K)=K*XSCNIN+0.5
13600 C JJ(K)=K*JSCNIN
13700 C FILL IN STOPPING TABLE
13800 C QSGRD=XSCNIN**2+JSCNIN**2
13900 C QDOTV=XSCNIN*XR+JSCNIN*YR
14000 C XHZERO=XR*QSGRD/QDOTV
14100 C DO 1330 K=NB1,NB2
14200 1330 ISTOP(K)=IP2+K*XHZERO+2.5
14300 C DETERMINE STOPPING SIGN.
14400 1350 IF (IP1-ISTOP(0)) 1350,1350,1360
14500 C ISTOPSG=1
14600 C GO TO 1370
14700 1360 ISTOPSG=-1
14800 C SET Q AND P.
14900 C FIND POINT Q RELATIVE TO NEW SEARCH COORDINATES.
15000 C START SEARCH THERE.
15100 1370 IP=IP1
15200 C YP1=JP1
15300 C IF (SLOPE.NE.0) GO TO 1380
15400 C IODIST=(IQ-IP1)*IINC
15500 C GO TO 1383
15600 C Let point QQ be the intersection of line P1P2 and
C a line perpendicular to P1P2 which passes through Q.
C Calculate the X coordinate of QQ.

```



```

17-JUL-70 0825 FOL10, SAV 1, MDK
15700 1380 X00=(YP1-J0+SLOPE*IP1+(1./SLOPE)*I0)
15800 1 / (SLOPE+(1./SLOPE))
15900 IQ0=X00+0.5
16000 IQDIST=(IQ0-IP1)*IINC
16100 IF (IQDIST) 1385, 1390, 1390
16200 N=0
16300 KQ=0
16400 WIDE=, TRUE,
16500 GO TO 13960
16600 N=IQDIST
16700 IP=IP1+N*IINC
16800 JP=YPI+N*YINC+0.5
16900 KQ=(J0-JP)*JSCNIN
17000 IF (J0.NE.I22SAV.OR.JQ.NE.J22SAV) GO TO 13963
17100 IF (IQDIST.LT.0) GO TO 13962
17200 LASTKQ=KQ
17300 GO TO 1500
17400 IF (SLOPE.EQ.0) GO TO 13964
17500 ISDIST=IQDIST
17600 GO TO 13960
17700 IF (SLOPE.NE.0) GO TO 13966
17800 LASTKQ=(J22SAV-JP1)*JSCNIN
17900 GO TO 1500
18000 XSS=(YP1-J22SAV+SLOPE*IP1+(1./SLOPE)*I22SAV)
18100 1 / (SLOPE+(1./SLOPE))
18200 ISS=XSS+0.5
18300 ISDIST=(ISS-IP1)*IINC
18400 JPS=YPI+ISDIST*YINC+0.5
18500 LASTKQ=(J22SAV-JPS)*JSCNIN
18600 GO TO 1500
18700
18800 C MOSTLY VERTICAL.
18900 HORIZ=, FALSE,
19000 XINC=-X1/ABS(Y1)
19100 JINC=ISIGN(1,JP2-JP1)
19200 ISCNIN=-JINC
19300 YSCNIN=XINC
19400 DO 1410 K=NB1,NB2
19500 II(K)=K*ISCNIN
19600 JJ(K)=K*YSCNIN+0.5
19700 OSGRD=ISCNIN**2+YSCNIN**2
19800 QDOTV=ISCNIN*XR+YSCNIN*YR
19900 YHZERO=YR*OSGRD/QDOTV
20000 DO 1430 K=NB1,NB2
20100 JSTOP(K)=JP2+K*YHZERO+0.5
20200 IF (JP1-JSTOP(0)) 1450, 1450, 1460
20300 JSTPSG=1
20400 GO TO 1470
20500 JSTPSG=-1
20600 XPI=IP1
20700 JP=JP1
20800 IF (X1.NE.0) GO TO 1480

```



```

20920 JDDIST=(JQ-JP1)*JINC
21220 GO TO 1483
21120 1480 YQ0=(SLOPE*JQ+(1./SLOPE)*JP1-[0+XP1])
21200 1 / (SLOPE+(1./SLOPE))
21300 JQ0=YQ0+0.5
21400 JDDIST=(JQ0-JP1)*JINC
21500 1483 IF (JDDIST) 1485,1490,1490
21600 1485 N=0
21700 KQ=0
21800 WIDE=,TRUE,
21900 GO TO 14960
22000 N=JDDIST
22100 IP=XP1+N*XINC+0.5
22200 JP=JP+N*JINC
22300 KQ=(IQ-IP)*ISCNIN
22400 IF (IQ,NE,I22SAV,OR,JQ,NE,J22SAV) GO TO 14963
22500 IF (JDDIST,LT,0) GO TO 14962
22600 LASTKQ=KQ
22700 CO TO 1500
22800 IF (X1,EQ,0) GO TO 14964
22900 JSDIST=JDDIST
23000 GO TO 14969
23100 14963 IF (X1,NE,0) GO TO 14966
23200 14964 LASTKQ=(I22SAV-[P1])*ISCNIN
23300 GO TO 1500
23400 14966 YSS=(SLOPE*J22SAV+(1./SLOPE)*JP1-[I22SAV+XP1])
23500 1 / (SLOPE+(1./SLOPE))
23600 JSS=YSS+0.5
23700 JSDIST=(JSS-JP1)*JINC
23800 IPS=XP1+JSDIST*XINC+0.5
23900 LASTKQ=(I22SAV-[PS])*ISCNIN
24000 GO TO 1500
24100 C INITIALIZE SEARCH,
24200 IF (KQ,GE,NB1,AND,KQ,LE,NB2) GO TO 1510
24300 KQ=NB1
24400 IF (KQ,GT,NB2) KQ=NB2
24500 WIDE=TRUE
24600 IF (.NOT,WIDE) GO TO 1525
24700 KS1=NB1
24800 KS2=NB2
24900 IBACK2=100
25000 IBACK1=100
25100 I=IQ
25200 1525 J=JQ
25300
25400 C*****
25500 C***** SEARCH *****
25600
25700 C UNTIL TRACE OF 0 CROSSES L
25800 IF (HORIZ) GO TO 1610
25900 1600 IF (JSTPSG*(J-JSTOP(KQ))) 1675,1675,3000
26000

```



```

26100 1610 IF (ISTPSG*(I-ISTOP(KQ))) 1650,1650,3000
26200 C
26300 SEARCH
26400 N=N+1
26500 IP=IP+IINC
26600 JP=YPI+N*YINC+0.5
26700 GO TO 1680
26800 N=N+1
26900 IP=XPI+N*XINC+0.5
27000 JP=JP+JINC
27100
27200 IF (NODPY) GO TO 1690
27300 CALL CLRPOG(UPOG)
27400
27500 C
27600 SEARCH ALONG S FROM S1 TO S2 FOR EDGE.
27700 DO 1720 K=KS1,KS2
27800 I=IP+I(K).
27900 J=JP+JJ(K)
28000 IF (NODPY) GO TO 1695
28100 CALL APOINT(G1(I),G2(J))
28200 CALL DRYOUT(UPOG)
28300 ISEGE=JFUNC(I,J,MDIRS)
28400 IF (ISEGE) 1900,1700,1900
28500 CONTINUE
28600 NO EDGE FOUND
28700 IBACK1=100
28800 IBACK2=100
28900 IF (WIDE) GO TO 1840
29000 SEARCH ALONG S FROM B1 TO S1-1 AND FROM S2+1 TO B2 FOR EDGE.
29100 DO 1820 K=NB1,NB2
29200 IF (K,GE,KS1.AND,K,LE,KS2) GO TO 1800
29300 I=IP+I(K)
29400 J=JP+JJ(K)
29500 IF (NODPY) GO TO 1795
29600 CALL APOINT(G1(I),G2(J))
29700 CALL DRYOUT(UPOG)
29800 ISEGE=JFUNC(I,J,MDIRS)
29900 IF (ISEGE) 1810,1800,1810
30000 CONTINUE
30100 WIDE=.TRUE.
30200 KS1=NB1
30300 KS2=NB2
30400 IF (ISEGE) 1820,1840,1820
30500 EDGE FOUND DURING WIDE SEARCH.
30600 KQ=K
30700 GO TO 2200
30800 NO EDGE FOUND.
30900 KQ=4*KQ/5
31000 I=IP+I(KQ)
31100 J=JP+JJ(KQ)
31200 GO TO 1600
31300 EDGE FOUND DURING NARROW SEARCH.

```



```

31300 1900 KQ=K
31400 IF (KQ.LT,NB1+2) GO TO 2000
31500 IBACK=LOAD(IP+1,(KQ-2),JP+JJ(KQ-2))
31600 IF (IBACK2.EQ.100) GO TO 1925
31700 IF (IARS(IBACK-IBACK1).GT.1 .OR.
31800 1 IARS(IBACK-IBACK2).GT.1) GO TO 2000
31900 C OK FOR NARROW SEARCH.
32000 GO TO 2000
32100 WIDE=.FALSE.
32200 IBACK2=IBACK1
32300 IBACK1=IBACK
32400 KS1=MAX0(KQ-4,NB1)
32500 KS2=MIN0(KQ+4,NB2)
32600 GO TO 2200
32700 IBACK2=IBACK1
32800 IBACK1=IBACK
32900 GO TO 2200
33000 C SET FOR WIDE SEARCH.
33100 2000 WIDE=.TRUE.
33200 IBACK2=100
33300 IBACK1=100
33400 KS1=NB1
33500 KS2=NB2
33600 GO TO 2230
33700 C RUN SIMPLE FILTER ON POINTS FOUND.
33800 IF (SFLAG) 2250,2210,2220
33900 C LASTKQ IS UNDEFINED, SET IT UP.
34000 2210 ASSIGN 1600 TO LABSP
34100 SFLAG=1
34200 C LASTKQ=KQ
34300 NEXT TWO VALUES USED ONLY IF DIRECTION IS CHANGED BETWEEN POINTS.
34400 I22SAV=I
34500 J22SAV=J
34600 GO TO 2400
34700 C NORMAL SITUATION. TEST KQ.
34800 2220 IF (IABS(KQ-LASTKQ).GT.3) GO TO 2230
34900 C NO BIG CHANGE IN KQ.
35000 C ASSERT: SFLAG=1, LABSP=1600.
35100 LASTKQ=KQ
35200 C NEXT TWO VALUES USED ONLY IF DIRECTION IS CHANGED BETWEEN POINTS.
35300 I22SAV=I
35400 J22SAV=J
35500 GO TO 2400
35600 C A BIG CHANGE IN KQ. HOLD I,J FOR A TEST ON THE NEXT POINT.
35700 2230 I22SAV=I
35800 J22SAV=J
35900 KQ22SV=KQ-LASTKQ
36000 C ASSERT: LABSP=1600.
36100 SFLAG=-1
36200 LASTKQ=KQ
36300 GO TO 1600

```


17-JUL-70 0825 FOL10.SAV 1.MDK PAGE 1-8

```

C      Previous I,J is being held.  Test new I,J to see what to do.
2250  IF (IABS(KQ-LASTKQ).GT.3) GO TO 2260
C      No big change in new KQ.
C      Store previous I,J.  Then new I,J.
36800  ASSIGN 2255 TO LABSP
36900  GO TO 2350
C      Now for new I,J.
37200  I=I22SAV
37100  J=J22SAV
2255  ASSIGN 1600 TO LABSP
37300  SFLAG=1
37400  LASTKQ=KQ
37500  GO TO 2400
37600
C      A big change in new KQ.  Compare signs of this change and
37700  the previous change (which was also big).
37800  IF (KQ22SV*(KQ-LASTKQ)) 2280,2265,2265
2260  Signs are the same.  Therefore previous I,J is okay
38200  and should be stored.  New I,J should be held as at
38100  2230 above.
C      ASSIGN 2270 TO LABSP
38200  GO TO 2350
2265  ASSIGN 1600 TO LABSP
38300  ASSERT: I22SAV,J22SAV = NEW COORDINATES.
2270  I=I22SAV
38400  J=J22SAV
38500  GO TO 2235
C      SIGNS OF TWO CONSECUTIVE BIG CHANGES ARE DIFFERENT.
38600  DELETE PREVIOUS POINT.  DISPLAY A "D" AT PREVIOUS POINT
C      IF DESIRED.
39100  IF (NODPY) GO TO 2290
2280  DISPLAY "D" AT I22SAV,J22SAV.
C      IF (ANYS) GO TO 2285
39200  ANYS=.TRUE.
39300  SPOG=IADPST(LSBUF)
39400  GO TO 2288
39500
2285  CALL SETPOG(SPOG)
2288  CALL DPYTXT(C1(I22SAV),G2(J22SAV),'D',1)
C      CALL DPYOUT(SPOG)
40100  NOW STORE NEW I,J.
40200  ASSERT: LABSP=1600.
C      SFLAG=1
2290  LASTKQ=KQ
C      NEXT TWO VALUES USED ONLY IF DIRECTION IS CHANGED BETWEEN POINTS.
40500  I22SAV=I
40600  J22SAV=J
40700  GO TO 2400
40800
C      STORE PREVIOUS I,J.
40900  ASSERT: LABSP contains proper exit.
C      ITEMP=1
2350  I=I22SAV
41200  I22SAV=ITEMP
41300  ITEMP=J
41400
41500

```



```

41620 J=J22SAV
41720 J22SAV=ITEMP
41800 GO TO 2420
41920
42020 STORE POINT (AND DISPLAY).
42100 CALL STORE(I,J,1)
42200 IF (ANYLIS) GOTO 2410
42300 ANYLIS=.TRUE.
42400 JLIST=LESTAB(I,J)
42500 WSTART=JLIST
42600 GO TO 2420
42720 JLIST=LRINS(JLIST,I,J)
42800 IF (IFDDPY.EQ.0) GO TO LABSP
42900 C COMPUTE OPY COORDS. OF POINT.
43000 NINEW=G1(I)
43100 NJNEW=G2(J)
43200 C INITIALIZE MPOG.
43300 IF (ANYM) GO TO 2450
43400 ANYM=.TRUE.
43500 NI=NINEW
43600 NJ=NJNEW
43650 CALL SETPOG(MPOG)
43720 CALL ALVECT(NI,NJ)
43820 C INITIALIZE NPOG.
43920 IF (NODPY) GO TO 2445
43950 CALL SETPOG(NPOG)
44020 CALL ALVECT(NI,NJ)
44100 ANYN=.FALSE.
44200 GO TO LABSP
44320 C COMPUTE NEW SLOPE.
44420 DXNEW=NINEW-NI
44500 IF (DXNEW.EQ.0) GO TO 2453
44600 SLOWNEW=(NJNEW-NJ)/DXNEW
44700 GO TO 2455
44820 SLOWNEW=9999.
44920 C IS SLOPE INITIALIZED?
45020 2455 IF (ANYN) GO TO 2460
45120 ANYN=.TRUE.
45200 GO TO 2483
45320 C TEST NEW SLOPE.
45400 2460 IF (SLOWNEW.EQ.SLOPEN) GO TO 2465
45520 TDIST=(TSLOPE*NINEW-NJNEW+TB)/TDENOM
45620 C GCONST = 1.5 * DISPLAY SCALE FACTOR.
45720 IF (ABS(TDIST),GE.GCONST) GO TO 2470
45820 C SET UP OLD NPOG.
45920 2465 IF (NODPY) GO TO 2485
46020 CALL SETPOG(NPOG)
46120 GO TO 2485
46200 C ERASE OLD NPOG.
46300 2470 IF (NODPY) GO TO 2473
46420 CALL HYDPOG(NPOG)
46500 C DISPLAY NEW MPOG.

```



```

46600 2473 CALL SETPOG(MPOG)
46722 CALL AVECT(NI,NJ)
46800 IF (NOOPY) GO TO 2480
46923 CALL DPYOUT(MPOG)
47000 C SET UP NEW NPOG.
47123 CALL CLRPOG(NPOG)
47223 CALL AIVECT(NI,NJ)
47323 C SET UP SLOPE PARAMETERS.
47423 2480 SLOPEN=SLONEW
47523 TSLOPE=SLONEW
47623 TB=NJ-TSLOPE*NI
47723 TDECOM=SQRT(TSLOPE**2+1)
47823 NI=NINEW
47923 NJ=NJNEW
48023 IF (NOOPY) GO TO 2490
48123 C DISPLAY NEW NPOG.
48223 CALL AVECT(NI,NJ)
48323 CALL DPYOUT(NPOG)
48423 2490 GO TO LABSP
48523
48623 C FINISHED WITH THIS P1 TO P2 SECTION.
48723 3000 IQ=I
48823 JQ=J
48923 WIDE=NXTWID.OR.WIDE
49023 RETURN
49123
49223 END

```



```

25323 DOONE (N4,N5,N6,N1,N2,N3)>
25423
25523 M1 M2 M3
25623 M4 M5
25723 M6 M7 M8
25823
25923 INTERN JEDGE3
26023 JEDGE3: 0
26123 SETM TFLAG# ;*****NEW
26223 MOVEI 0,TEMP
26323 BLT 0,TEMP+LASTAC
26423 JSA 17,SPLOAD ;RETURNS WITH A1=POINTER TO I-1,J
26523 EXTERN SPLOAD ; A2=LINLEN
26623 LDR 0,CPOINT 6,1,53
26723 CAIE 0,44 ;DOES POINTER IN AC1=448B00,,A ?
26823 JRST PNTOK ;NO.
26923 TLZ 1,770000. ;YES. SET TO 008B00,,A-1
27023 SOS 1
27123 PNTOK: MOVE 0,1
27223 SUB 0,2 ;PREVIOUS LINE A0=A1-LINLEN
27323 LGS M1,0
27423 ILDB M2,0
27523 ILDB M3,0
27623 MOVE 0,1 ;CURRENT LINE
27723 LDB M4,0
27823 IBP 0
27923 ILDB M5,0
28023 ADD 1,2 ;NEXT LINE A1=A1+LINLEN
28123 LDB M6,1
28223 ILDB M7,1
28323 ILDB M8,1
28423 SKIPN NEWWAY
28523 JRST X99 ;OLD METHOD. NEWWAY.EQ.0
28623 JRST NEW99 ;NEW METHOD. NEWWAY.NE.0
28723 NEWWAY: 0
28823
28923 INTERN JEDGE5
29023 JEDGE5: 0
29123 INEWER***** SETM TFLAG ;*****NEW
29223 MOVEI 0,TEMP
29323 BLT 0,TEMP+LASTAC
29423 JSA 17,SPLOAD ;RETURNS WITH A1=POINTER TO I-1,J
29523 EXTERN SPLOAD ; A2=LINLEN
29623 JUMPL 1,X69
29723 ADD 1,0040000000000000 ;BYTE POSITION IS 4 TO 34(OCTAL)
29823 JRST PNTOK5 ;ADD 4 TO IT.
29923 TLZ 1,7000000 ;BYTE POSITION IS 40 OR 44(OCTAL). SET TO
30023 SOS 1 ;00 OR 04, ADDR TO ADDR-1.
30123 PNTOK5: MOVE 0,1
30223 SUB 0,2 ;PREVIOUS LINE A0=A1-LINLEN
30323 SUB 0,2
30423 LDB M1,0

```



```

10500  ISB 0
10600  ILDB M2,0
10700  ISB 0
10800  ILDB M3,0
10900  MOVE 0,1
11000  LDB M4,0
11100  ISB 0
11200  ISB 0
11300  ISB 0
11400  ILDB M5,0
11500  ADD 1,2 ;NEXT LINE 01=A1+LINLEN
11600  ADD 1,2
11700  LDB M6,1
11800  ISB 1
11900  ILDB M7,1
12000  ISB 1
12100  ILDB M8,1
12200  MOVE MDIRS,02(16)
12300  SETM VALUE
12400
12500  TRNE MDIRS,1
12600  JRST 001
12700  TRNE MDIRS,2
12800  JRST 002
12900  TRNE MDIRS,4
13000  JRST 004
13100  TRNE MDIRS,08
13200  JRST 008
13300  MOVSI LASTAC,TEMP+1
13400  HRR1 LASTAC,1
13500  BLT LASTAC,LASTAC
13600  JRA 16,3(16)
13700
13800  DOIT (1,M1,M4,M6,M3,M5,M8,TEST2)
13900  DOIT (2,M1,M2,M3,M6,M7,M8,TEST4)
14000  DOIT (4,M1,M2,M4,M5,M7,M8,TEST8)
14100  DOIT (08,M2,M3,M5,M4,M6,M7,EXIT)
14200
14300  BLOCK 20
14400
14500  ; THRESHOLD FOR DOSIM
14600  DOSIM (BIT,N1,N2,NEXT)
14700  <MOVE K1,N1
14800  SUB K1,N2
14900  MOVMS K1
15000  CAIG K1,TDS
15100  JRST NEXT
15200  ORI VALUE,BIT
15300  JUMPL MDIRS,NEXT
15400  JRST EXIT>
15500
15600  NEW99: MOVE MDIRS,02(16)

```



```

15700      SETM VALUE
15820
15900      TRNE MDIRS,1
16020      JRST D01S
16120      TEST2S: TRNE MDIRS,2
16220      JRST D02S
16320      TEST4S: TRNE MDIRS,4
16420      JRST D04S
16520      TEST8S: TRNE MDIRS,*D8
16620      JRST D08S
16700
16820      D01S:  DOSIM (1,M4,M5,TEST2S)
16920      D02S:  DOSIM (2,M2,M7,TEST4S)
17020      D04S:  DOSIM (4,M1,M8,TEST8S)
17120      D08S:  DOSIM (*D8,M3,M6,EXIT)
17200
17300
17400      END

```


3781
1970
K
Stack

**Stanford University Libraries
Stanford, California**

Return this book on or before date due.

JULY 18, 1976

JUN 15 1978

MAR 28 1992-10
LIBRARY OF THE

AUG 15 1993

I.L.L.

MAY 11 1993-11

